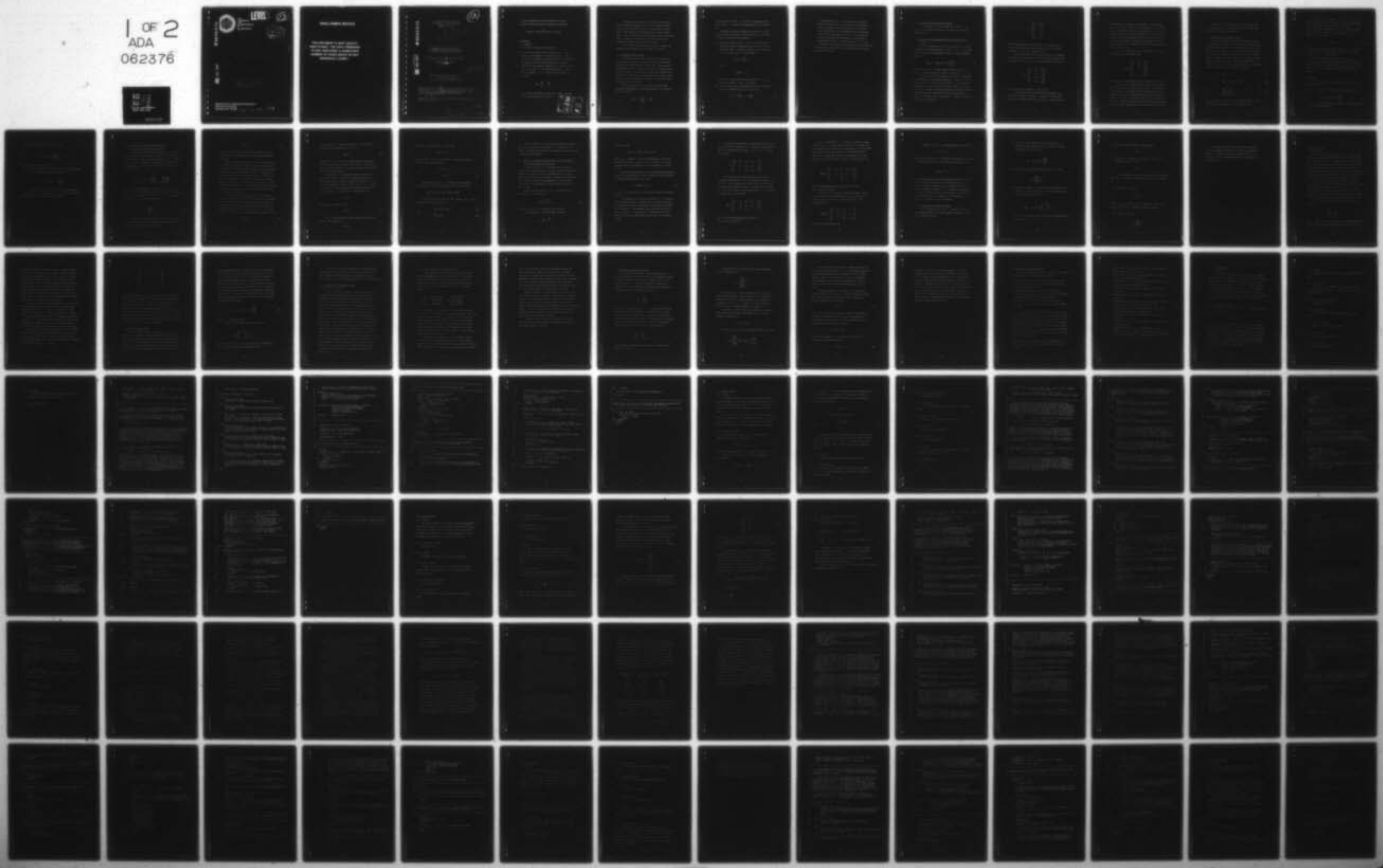
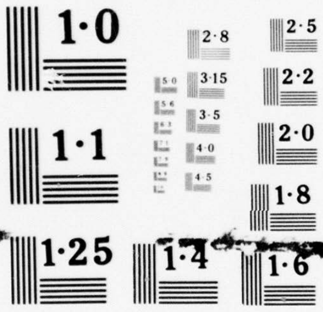



1·0
2·8
2·5
5·0
5·6
6·3
3·15
2·2
3·5
2·0
4·0
4·5
1·1
1·8
1·25
1·4
1·6

13

Systems
Optimization
Laboratory

Department of Operations Research
Stanford University
Stanford, CA 94305

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
Stanford University
Stanford, California

(9) Technical rept.,

(14) SOL-78-8

(6)

FORTRAN SUBROUTINES TO SOLVE THE LINEAR LEAST-SQUARES PROBLEM AND COMPUTE THE COMPLETE ORTHOGONAL FACTORIZATION

by

(10) Margaret H. Wright ~~and~~ Steven C. Glassman

TECHNICAL REPORT SOL 78-8
(11) April 1978

(12) 141p.

(15)
Research and reproduction of this report were partially supported by the Department of Energy Contract EY-76-S-03-0326 PA #18; the Office of Naval Research Contract N00014-75-C-0267, the National Science Foundation ~~Grants~~ MCS76-20019 A01 and ENG 77-06761.
NSF-

FORTRAN SUBROUTINES TO SOLVE THE LINEAR LEAST-SQUARES PROBLEM AND COMPUTE THE COMPLETE ORTHOGONAL FACTORIZATION

by

Margaret H. Wright and Steven C. Glassman

1. Introduction

1.1. Overview

The topics of interest in this report are:

(i) The linear least-squares problem -- given a real m by n matrix A and an m-vector b, find an n-vector x such that $\|Ax - b\|_2^2$ is a minimum; this problem includes the solution of non-singular, over- and under-determined linear systems;

(ii) computation of the complete orthogonal factorization of a real matrix A of rank r -- find orthogonal matrices Q and V, and a non-singular r by r upper triangular matrix R, such that:

$$QAV = \begin{bmatrix} R & 0 \\ & 0 \end{bmatrix},$$

where either or both blocks of zeros may be absent, depending on the relative values of m, n, and r.

The software to be described is a modular set of portable Fortran subroutines designed to carry out the computations necessary to solve (i) and (ii), and also to perform several related tasks. These routines have already been used in several applications -- e.g., in large-scale linear programming (Perold and Dantzig, 1978), in solving ill-conditioned linear systems arising from stochastic processes, and in quadratic programming.

If the reader is interested only in using the software, the descriptions and documentation are contained in Section 6.

1.2. Householder Transformations

The techniques used to solve (i) and (ii) are based on the construction of a sequence of orthogonal transformations designed to reduce the original matrix to a special form. The advantages of this approach in terms of elegance, efficiency and numerical stability are well known. The theory and procedures were popularized primarily by G.H. Golub, and are explained in detail in numerous references (e.g., Lawson and Hanson, 1974; Stewart, 1973); only a brief description will be given here.

For any non-zero vector u, we define the corresponding *Householder transformation* (or *Householder matrix*) as an elementary matrix of the form:

$$H(u) \equiv I - \frac{2uu^T}{\|u\|_2^2} \equiv I - \frac{uu^T}{\beta} .$$

In this context, the vector u is called the Householder vector, and the scalar β is termed the scaling factor for the transformation.

Householder matrices are symmetric and orthogonal (so that Euclidean length is preserved by their application). In addition, they have the following useful properties:

(a) for any two distinct vectors a and b of equal Euclidean length, there exists a Householder matrix that will transform one into the other. In order to construct such a transformation H, we seek a vector u that satisfies:

$$Ha = (I - \frac{uu^T}{\beta})a = b \quad ,$$

so that

$$(- \frac{u^T a}{\beta})u = b - a \quad . \tag{1}$$

From (1), u must be a vector in the direction $(b - a)$, and is non-zero because a and b are distinct.

(b) The vector that results from applying a Householder transformation is of a special form, since for any vector c:

$$Hc = (I - \frac{uu^T}{\beta})c = c - u(\frac{u^T c}{\beta}) \quad . \tag{2}$$

The transformed vector Hc is thus given by the difference between the original vector and a multiple of the Householder vector. Consequently, the transformed vector is identical to the original vector in all components where the Householder vector is zero; furthermore, the vector c is not altered at all by the Householder transformation if it is orthogonal to the Householder vector, i.e., the inner product u^Tc is zero.

These properties of Householder transformations can be exploited to reduce a general real matrix to a form that permits easy solution of problems (i) and (ii).

2. The Full-Rank Linear Least-Squares Problem

In this section, it will be assumed that $\mathrm{rank}(A) = n$, so that $m \geq n$. We defer until Section 4 the complex issue of determining the rank.

2.1. Reduction to Upper Triangular Form by Transformation From the Left

Because of properties (a) and (b), it is possible to construct a sequence of Householder matrices -- $H_1, H_2, \ldots, H_n$ -- to be applied to the original matrix on the left to reduce it to upper triangular form, i.e.,

$$H_n H_{n-1} \cdots H_2 H_1 A \equiv QA = \hat{R} = \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} \tag{3}$$

where $\bar{R}$ is an n by n upper triangular matrix, and Q is an m by m orthogonal matrix (the product of the matrices $H_n \cdots H_1$).

The first step of this reduction is to construct a Householder transformation H_1 to annihilate components 2 through m of the first column of A. From (1), it follows that u_1, the vector corresponding to H_1, will be equal to the first column of A, except for the first component. Because Householder transformations preserve Euclidean length, the transformed first column will be given by $[r_{11}\ 0 \cdots 0]^T$, where $|r_{11}| = (a_{11}^2 + a_{21}^2 + \cdots + a_{m1}^2)^{1/2}$. Using (1), the vector u_1 is given by:

$$u_1 = \begin{bmatrix} a_{11} - r_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} .$$

To avoid cancellation error in computing u_1, the sign of r_{11} is chosen to be opposite that of the original a_{11}; thus, the first component of u_1 is $(a_{11} + \text{sign}(a_{11})|r_{11}|)$.

After application of H_1, the first column of the transformed matrix has the form of the first column of an upper triangular matrix. For example, if $m = 4$, $n = 3$, the transformed matrix is given by:

$$H_1A = \begin{bmatrix} r_{11} & \bar{a}_{12} & \bar{a}_{13} \\ 0 & \bar{a}_{22} & \bar{a}_{23} \\ 0 & \bar{a}_{32} & \bar{a}_{33} \\ 0 & \bar{a}_{42} & \bar{a}_{43} \end{bmatrix}$$

and, in general, all elements of A are altered.

In constructing the second Householder transformation H_2, the aim is to annihilate components 3 through m of the second column, while leaving the first column unaltered. From (2), this can be accomplished by setting the first component of the Householder vector u_2 to zero, for then u_2 is orthogonal to the first column of H_1A. Furthermore, because the first component of u_2 is zero, the first row of H_1A will not be altered by H_2. Since the first row and column remain unchanged, they can be ignored; in essence, the second transformation is applied to the "remaining matrix" of $(m - 1)$ rows and $(n - 1)$ columns that results from omitting the first row and column of H_1A.

After H_2 is applied to H_1A, the second column is in the desired form. With the example above,

$$H_2H_1A = \begin{bmatrix} r_{11} & \bar{a}_{12} & \bar{a}_{13} \\ 0 & r_{22} & \bar{\bar{a}}_{23} \\ 0 & 0 & \bar{\bar{a}}_{33} \\ 0 & 0 & \bar{\bar{a}}_{43} \end{bmatrix}$$

where r_{22} and the doubly barred elements have been altered by H_2.

This process is continued until the matrix has been reduced to upper triangular form. Each step may be viewed as transforming the "first" column of a successively smaller remaining matrix, since by construction the $(k + 1)$-st transformation does not alter rows or columns 1 through k. It should be noted that the assumption of full rank is essential to ensure that the first column of the remaining matrix is never identically zero at any stage of the reduction.

Details of the actual computation, operation counts, and data structures used in constructing, applying and storing these transformations are given in Section 6.

2.2. Solution of the Least-Squares Problem

After A has been reduced to upper triangular form by the process described in Section 2.1, the matrix $\bar{R}$ in (3) will be non-singular (because rank(A) = n). The unique solution of the full-rank least-squares problem is then computed as follows. The matrix Q (the product of the Householder matrices $H_n \cdots H_1$) reduces A to upper triangular form from the left and is orthogonal. Because Euclidean length is preserved by orthogonal transformation, the least-squares residual of the problem transformed by Q is equal in length to the residual of the original problem; thus:

$$\|Ax - b\|_2 = \tag{4a}$$

$$\|Q(Ax - b)\|_2 = \|\hat{R}x - Qb\|_2 = \tag{4b}$$

$$\left\| \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} x - \begin{bmatrix} \bar{b} \\ \tilde{b} \end{bmatrix} \right\|_2 , \tag{4c}$$

where the vector Qb has been partitioned into its first n components $(\bar{b})$ and remaining $(m - n)$ components $(\tilde{b})$.

Because rows $n + 1$ through m of $\hat{R}$ are zero, it is clear from inspection of (4) that the residual vector of the transformed problem will be minimized when the first n components of $\hat{R}x$ are equal to the first n components of Qb, i.e., a minimum residual will be attained for the vector x that satisfies the n by n non-singular linear system:

$$\bar{R}x = \bar{b} \quad . \tag{5}$$

The minimum residual will have length $\|\tilde{b}\|_2$, since the residual of the transformed problem (4c) is zero in the first n components, by construction of x to satisfy (5).

The orthogonality of the transformations used to reduce A to triangular form is crucial in solving the least-squares problem; the solution of the transformed problem in (4) is equivalent to the solution of the original problem only because Euclidean length is preserved.

2.3. Summary

In the full-rank case, the linear-least squares problem is solved as follows:

(a) construct a sequence of Householder transformations such that

$$H_n \cdots H_2H_1A \equiv QA = \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix},$$

where $\bar{R}$ is upper-triangular; the result is sometimes described as the "QR factorization" of A.

(b) transform the right-hand side, i.e., form

$$H_n \cdots H_2 H_1 b \equiv Qb = \begin{bmatrix} \bar{b} \\ \tilde{b} \end{bmatrix} \begin{matrix} \}\ n \\ \}\ m-n \end{matrix} \quad ;$$

(c) solve the triangular system $\bar{R}x = \bar{b}$;

(d) if desired, compute the residual vector for the original problem by back-transforming the transformed residual:

$$Ax - b = Q^T[QAx - Qb] = Q^T \begin{bmatrix} 0 \\ \tilde{b} \end{bmatrix} \quad .$$

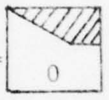
Step (a) needs to be carried out only once for a given matrix A, after which the least-squares problem can be solved for different right-hand sides by repeating steps (b) through (d).

3. The Rank-Deficient Linear Least-Squares Problem

3.1. Non-Uniqueness; the Minimum-Length Solution

If the matrix A has deficient column rank, the linear least-squares problem is more complicated. If rank $(A) = r$, where $r < n$, we can assume for purposes of exposition that the first r columns of A are linearly independent. After applying r Householder transformations to A on the left, as described in Section 2.1, the result will be:

$$H_r \cdots H_1 A \equiv QA = \hat{R} = \begin{array}{c@{}l} \overbrace{\phantom{\bar{R}}}^{n} & \\ \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} & \begin{array}{l} \} \ r \\ \} \ m-r \end{array} \end{array} =$$ (6a)



where $\bar{R}$ is upper trapezoidal, since columns $(r + 1)$ through n of A are by assumption linearly dependent on the first r columns (the block of zeros below $\bar{R}$ is absent if $m = r$). Let the transformed vector Qb be similarly partitioned:

$$Qb = \begin{bmatrix} \bar{b} \\ \tilde{b} \end{bmatrix} \begin{array}{l} \} \ r \\ \} \ m-r \end{array} . \qquad (6b)$$

From (6), it can be seen, as in the full-rank case, that a minimum-length residual $\|Ax - b\|_2$ will be attained for any vector x that satisfies the set of r equations:

$$\bar{R}x = \bar{b} \quad . \tag{7}$$

Because $\bar{R}$ has fewer rows than columns, the solution to (7) is not unique, and thus the solution to the least-squares problem is not unique.

Among all vectors satisfying (7), it is often desired to find the one with minimum Euclidean length. There are various ways to characterize the minimum-length least-squares solution; a full discussion is given in Peters and Wilkinson (1970). The properties of interest here derive from the observation that the r rows of the upper trapezoidal matrix $\bar{R}$ are linearly independent, and span a subspace of dimension r in n-space, of all vectors that are linear combinations of the rows of $\bar{R}$. The minimum-length vector that solves the least-squares problem is the unique vector satisfying (7) that lies entirely in this subspace.

To see why this is true, assume that the r columns of a matrix $\bar{V}$ form a basis for this subspace. The particular representation of $\bar{V}$ is not relevant initially (for example, the rows of $\bar{R}$ clearly comprise such a basis); additional discussion is given in Section 3.2. By definition of a basis, any vector $\bar{x}$ in this subspace may be written as a linear combination of the columns of $\bar{V}$, i.e.,

$$\bar{x} = \bar{V}\bar{w} \tag{8a}$$

for some r-vector $\bar{w}$. To show that such an $\bar{x}$ will satisfy (7), we substitute the expression for $\bar{x}$ into (7):

$$\bar{R}\bar{V}\bar{w} = \bar{b} \quad . \tag{8b}$$

The matrix $\bar{R}\bar{V}$ is r by r, and is non-singular by definition of $\bar{V}$ as a basis; hence, $\bar{w}$ is unique. Thus, when $\bar{w}$ is the solution of (8b), $\bar{x} = \bar{V}\bar{w}$ satisfies (7), and yields a minimum-length residual for the least-squares problem.

To verify that $\bar{x}$ has minimum Euclidean length among all vectors satisfying (7), we consider a complementary subspace to that mentioned above -- namely, the subspace of dimension $(n - r)$ of vectors orthogonal to the rows of $\bar{R}$. Let the $(n - r)$ columns of the matrix $\tilde{V}$ form a basis for this complementary subspace, so that every vector z for which $\bar{R}z = 0$ may be written as:

$$z = \tilde{V}y$$

for some $(n - r)$-vector y, and

$$\bar{V}^T\tilde{V} = 0 \quad . \tag{9}$$

Let $\tilde{x}$ be any vector that yields a minimum-length residual; then $\tilde{x}$ must satisfy (7):

$$\bar{R}\tilde{x} = \bar{b} \quad .$$

Because $\bar{x}$ also satisfies (7), it follows that

$$\bar{R}(\tilde{x} - \bar{x}) = 0 \quad .$$

Thus, the vector $(\tilde{x} - \bar{x})$ is orthogonal to the rows of $\bar{R}$, and, as indicated above, may be written as:

$$\tilde{x} - \bar{x} = \tilde{V}y \quad ,$$

or

$$\tilde{x} = \bar{x} + \tilde{V}y \quad . \tag{10}$$

Consequently, <u>every</u> vector $\tilde{x}$ satisfying (7) may be written in the form (10). Consider the Euclidean length of such a vector:

$$\|\tilde{x}\|_2^2 = \|\bar{x} + \tilde{V}y\|_2^2 = \|\bar{x}\|_2^2 + 2\bar{x}^T\tilde{V}y + \|\tilde{V}y\|_2^2 \quad .$$

It follows from (9) that the term $\bar{x}^T\tilde{V}y$ vanishes, since $\bar{x} = \bar{V}\bar{w}$; the expression for $\|\tilde{x}\|_2^2$ then becomes:

$$\|\tilde{x}\|_2^2 = \|\bar{x}\|_2^2 + \|\tilde{V}y\|_2^2 \quad , \tag{11a}$$

so that

$$\|\tilde{x}\|_2^2 \geq \|\bar{x}\|_2^2 \; . \tag{11b}$$

Since the columns of $\tilde{V}$ are linearly independent by construction, equality holds in (11b) only when $y = 0$. Therefore, the vector $\bar{x}$ defined by (8a) and (8b) is the unique least-squares solution of minimum Euclidean length.

3.2. Reduction of an Upper Trapezoidal Matrix to Upper Triangular Form by Transformation From the Right

As shown in Section 3.1, the minimum-length least-squares solution can be computed using a set of basis vectors for the subspace spanned by the rows of the upper trapezoidal matrix $\bar{R}$. Through application of another special sequence of Householder transformations, it is possible to construct an orthogonal basis for this subspace by reducing $\bar{R}$ from the right to an upper triangle followed by a block of zeros.

Suppose that there exists an n by n orthogonal matrix V that reduces $\bar{R}$ to this form, i.e.:

$$\bar{R}V = [\overbrace{R}^{r} \;\vdots\; \overbrace{0}^{n-r}] , \tag{12}$$

where R is an r by r non-singular upper triangle.

If the columns of V are partitioned accordingly:

$$V = [\overbrace{\bar{V}}^{r} \;\vdots\; \overbrace{\tilde{V}}^{n-r}] ,$$

then it holds that

$$\bar{R}[\bar{V} \mid \tilde{V}] = [\bar{R}\bar{V} \mid \bar{R}\tilde{V}] = [R \mid 0] \quad .$$

Hence, the r columns of $\bar{V}$ form an orthogonal basis for the space spanned by the rows of $\bar{R}$, and the $(n - r)$ columns of $\tilde{V}$ form an orthogonal basis for the subspace of vectors orthogonal to the rows of $\bar{R}$.

From the results in Section 3.1, the minimum-length least-squares solution $\bar{x}$ may be written as $\bar{x} = \bar{V}\bar{w}$. Equation (7), which specifies the minimum-residual property, then becomes:

$$\bar{R}\bar{x} = \bar{R}\bar{V}\bar{w} = R\bar{w} = \bar{b} \quad , \tag{13}$$

so that $\bar{w}$ is the solution of a linear system involving the triangular matrix R.

The orthogonality of the chosen representation of $\bar{V}$ guarantees that the condition number of the matrix $\bar{R}\bar{V}$ is no greater than that of $\bar{R}$; thus, the "natural" conditioning of the problem is not altered by an orthogonal $\bar{V}$. This favorable result would not hold for some other choices of $\bar{V}$ -- for example, if $\bar{V}$ were taken as $\bar{R}^T$, the matrix in (13) would be $\bar{R}\bar{R}^T$, whose condition number is the square of $\text{cond}(\bar{R})$.

The process of constructing an orthogonal matrix V that satisfies (12) involves the definition of r Householder transformations, and is best illustrated by an example. Let $n = 5$, $r = 3$, so that $\bar{R}$ is given by:

$$\bar{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ 0 & r_{22} & r_{23} & r_{24} & r_{25} \\ 0 & 0 & r_{33} & r_{34} & r_{35} \end{bmatrix} .$$

The first step of the reduction involves constructing a Householder transformation -- $\bar{H}_3$ -- that annihilates elements 4 and 5 in row 3. Since elements 1 and 2 of row 3 are already zero, the corresponding Householder vector will be non-zero only in positions 3, 4, and 5, and so will not alter columns 1 and 2 of $\bar{R}$. The result of applying $\bar{H}_3$ is the following:

$$\bar{R}\bar{H}_3 = \begin{bmatrix} r_{11} & r_{12} & \bar{r}_{13} & \bar{r}_{14} & \bar{r}_{15} \\ 0 & r_{22} & \bar{r}_{23} & \bar{r}_{24} & \bar{r}_{25} \\ 0 & 0 & \bar{r}_{33} & 0 & 0 \end{bmatrix} ,$$

where the barred elements have been altered, and $|\bar{r}_{33}| = (r_{33}^2 + r_{34}^2 + r_{35}^2)^{1/2}$.

Next, a transformation $\bar{H}_2$ is required to annihilate elements 4 and 5 of row 2. In order for the form of row 3 to be retained, the inner product of row 3 and the vector corresponding to $\bar{H}_2$ must vanish, which implies that the third component of the Householder vector must be zero. Consequently, the vector corresponding to $\bar{H}_2$ will have non-zeros only in positions 2, 4, and 5, and the effect of $\bar{H}_2$ is:

$$\bar{R}\bar{H}_3\bar{H}_2 = \begin{bmatrix} r_{11} & \bar{\bar{r}}_{12} & \bar{r}_{13} & \bar{\bar{r}}_{14} & \bar{\bar{r}}_{15} \\ 0 & \bar{\bar{r}}_{22} & \bar{r}_{23} & 0 & 0 \\ 0 & 0 & \bar{r}_{33} & 0 & 0 \end{bmatrix}$$

where the doubly barred elements have been altered, and
$|\bar{\bar{r}}_{22}| = (r_{22}^2 + \bar{r}_{24}^2 + \bar{r}_{25}^2)^{1/2}$.

Finally, $\bar{H}_1$ is constructed to annihilate elements 4 and 5 in the first row; the vector corresponding to $\bar{H}_1$ must have zeros in positions 2 and 3, in order not to affect rows 2 and 3. The ultimate result is:

$$\bar{R}\bar{H}_3\bar{H}_2\bar{H}_1 = \begin{bmatrix} \bar{\bar{\bar{r}}}_{11} & \bar{\bar{r}}_{12} & \bar{r}_{13} & 0 & 0 \\ 0 & \bar{\bar{r}}_{22} & \bar{r}_{23} & 0 & 0 \\ 0 & 0 & \bar{r}_{33} & 0 & 0 \end{bmatrix}$$

which is the desired reduced form.

In summary, the n by n orthogonal matrix V that satisfies:

$$\bar{R}V = [R \mid 0]$$

is given by the product of r Householder transformations, constructed in a backward sweep over the rows as described, and may be written as

$$V = \bar{H}_r \bar{H}_{r-1} \cdots \bar{H}_1 \quad .$$

It should be emphasized that these transformations do not have the same structure as those used in the reduction from the left: in general, the vector corresponding to H_i (applied on the left) has zeros in components 1 through $(i - 1)$, and non-zeros in components i through m; whereas the vector corresponding to $\bar{H}_i$ (applied on the right) has non-zeros in component i and components $r + 1$ through n, and zeros in all other components.

3.3. Solution of the Least-Squares Problem

If it is assumed that the first r columns of the matrix A are linearly independent, the minimum-length least-squares solution may be computed as follows:

(a) Reduce A to upper trapezoidal form by application of r Householder transformations on the left, using the procedure described in Section 2.1, so that

$$H_r \cdots H_1 A \equiv QA = \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} \begin{matrix} \} \, r \\ \end{matrix} .$$

(b) Transform the right-hand side by applying Q, i.e., form

$$Qb = \begin{bmatrix} \bar{b} \\ \tilde{b} \end{bmatrix} \begin{matrix} \} \, r \\ \} \, m-r \end{matrix}$$

(c) Construct a sequence of r Householder transformations to be applied on the right to reduce $\bar{R}$ to upper triangular form, as described in Section 3.2, so that

$$QA\bar{H}_r \cdots \bar{H}_1 \equiv QAV = \begin{array}{c} \begin{matrix} \overbrace{}^{r} & \overbrace{}^{n-r} \end{matrix} \\ \begin{bmatrix} R & 0 \\ 0 & \end{bmatrix} \end{array} ,$$

where V is the product of the second set of Householder transformations.

(d) Solve the non-singular upper triangular system

$$R\bar{w} = \bar{b} \quad .$$

(e) Transform $\bar{w}$ by applying $\bar{V}$, the first r columns of V, yielding the solution $\bar{x}$ as

$$\bar{x} = \bar{V}\bar{w} \quad .$$

If the minimum-length solution is not required, a least-squares solution can be computed by carrying out steps (a) and (b), followed by:

(c') partition $\bar{R}$ as follows:

$$[\overbrace{\tilde{R}}^{r} \quad \overbrace{S}^{n-r}],$$

so that $\tilde{R}$ is the left-most r by r submatrix; $\tilde{R}$ must be non-singular if the first r columns are linearly independent.

(d') solve $\tilde{R}\tilde{x} = \bar{b}$, and let

$$\bar{x} = \begin{bmatrix} \tilde{x} \\ 0 \end{bmatrix} \quad .$$

The solution obtained in this way will satisfy equation (7), and is the least-squares solution of the problem corresponding to the first r columns of A and the given right-hand side. In general, it is not the minimum-length solution corresponding to all n columns of A.

4. Estimation of Rank

4.1. Difficulties

The procedures described in Section 3 for the rank-deficient least-squares problem assumed that the rank of A was known *a priori* to be r, and that the first r columns of A were linearly independent. These assumptions are not realistic, and were introduced only for simplicity of presentation. In attempting to solve a least-squares problem where the matrix A is of unknown rank, it is obviously necessary to estimate the rank during the course of the computation, and thereby to determine a set of linearly independent columns.

Unfortunately, the definition of "rank" in the context of computation with floating point arithmetic and inaccurate data depends on the problem. The question can never be resolved in a specific case without making an explicit judgment about the scaling, i.e., a determination as to which quantities can be considered as "negligible" (for an excellent discussion, see Golub, Klema, and Stewart, 1976).

As an illustration of the complexity of the issue, consider the matrix

$$\begin{bmatrix} 1 & 1 \\ 0 & \varepsilon \end{bmatrix},$$

where ε is not zero, but is small relative to unity. Mathematically, the two columns are linearly independent, and the matrix has rank 2.

In practice, however, the second vector may be a computed version of the first, so that numerically the two columns should be considered "equivalent", even though one is not an exact multiple of the other; in this event, the matrix has "rank" 1. Thus, the decision as to whether these two vectors are linearly independent depends on whether or not the value of ε is "negligible"; even in the simplest 2 by 2 case, the issue of rank can be resolved only by considering the nature of the underlying problem, and the origins of the data. For larger values of n, a satisfactory automatic technique for determining the rank of a matrix is still more complicated.

With exact arithmetic, linear dependence among a set of columns would reveal itself during the reduction to upper triangular form described in Section 2.1. If the $(k + 1)$-st column were a linear combination of the previous k columns, components $k + 1$ through m of the transformed dependent column (the "remaining column") would be exactly zero. With finite precision computation, one might accordingly hope that the norm of a remaining column would be "small" if that column were "nearly" linearly dependent on the previous columns.

Unfortunately, this hope is not realized, since it is equivalent to expecting that an ill-conditioned triangular matrix will have at least one small diagonal element. Triangular matrices exist that have no "small" diagonal elements, yet are arbitrarily badly conditioned -- for example, the famous n by n matrix:

$$U_n = \begin{bmatrix} 1 & -1 & -1 & \cdots & -1 \\ & 1 & -1 & \cdots & -1 \\ & & 1 & & \\ & 0 & & \ddots & \\ & & & & 1 \end{bmatrix},$$

whose condition number is of order 2^{n-1}, and which, even for moderate values of n, could reasonably be termed "numerically rank-deficient".

The implication of this example for the estimation of rank during triangular reduction is clear: if one were reducing U_n to upper triangular form, no transformations would be necessary (since it is already an upper triangle), and no indication of the near linear dependence among the columns would be given. Consequently, a strategy based on the expectation of a "small" remaining column in the presence of near linear dependence can not be guaranteed.

4.2. Column Interchange Strategy

Despite the rather artificial example given in Section 4.1, numerical linear dependence is almost invariably revealed in practice by a "small" remaining column during the reduction from the left (see further remarks that strengthen this observation in Golub, Klema, and Stewart, 1976). Thus, an obvious strategy for estimating rank and selecting a set of linearly independent set of columns is to carry out column interchanges during the reduction from the left, and to reduce at the k-th step the remaining column of "largest" norm, in order to move the "small" columns to the end. This strategy is discussed in Peters and Wilkinson (1970) and Lawson and Hanson (1974), and is sometimes called "column pivoting". It is not guaranteed to move the best-conditioned set of columns to the "front" (left) of the matrix, but in general it tends to allow the "most independent" columns to be processed first. The effect of interchanging columns is to introduce a permutation matrix on the right of the matrix A in (6a); the hoped-for configuration then becomes:

$$H_r \cdots H_1 AP \equiv QAP = \hat{R} = \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} \tag{14}$$

where P is a permutation matrix.

In practice, the computed version of (14) will be

$$QAP = \begin{array}{c} \begin{array}{cc} \overbrace{\ }^{r} & \overbrace{\ }^{n-r} \end{array} \\ \begin{bmatrix} \tilde{R} & S \\ 0 & E \end{bmatrix} \begin{array}{l} \} \; r \\ \} \; m-r \end{array} \end{array} ,$$

where $\|E\|$ is "small". A detailed discussion of the significance of E is given in Golub, Klema, and Stewart (1976).

In solving the least-squares problem with a column interchange strategy, steps (e) and (d') of the procedures given in Section 3.3 must be expanded to include application of the permutation matrix P to the solution $\bar{x}$ (i.e., appropriate components of $\bar{x}$ are interchanged).

4.3. Considerations in Estimating the Rank

4.3.1. "Size" of a column

The idea of estimating rank with a column interchange strategy to reduce the "largest" remaining column at each step leads to an interesting question: what should be the definition of the "size" of a column? The most obvious measure of the size of a vector is simply its Euclidean length; but since the Householder reduction from the left acts on each column independently, this definition is appropriate only if one assumes that the matrix to be reduced has been scaled initially so that the same definition of "small" applies uniformly to all columns. In many cases, however, each column represents observations of a particular variable, and the values of the variables may not display a constant scaling with respect to one another -- for example, the average magnitude of one variable might be 10^{-2}, while another might be of order 10^{5}. In such an instance, it would be wrong to regard the column corresponding to the first variable as seven orders of magnitude "smaller" than the column corresponding to the second variable. Clearly, it is necessary to exercise care in the definition of "size" in order to use a column interchange strategy to estimate rank. This topic will be discussed in more detail in Section 6.4.11.

4.3.2. Factors influencing the estimate of rank

In many applications it makes practical sense to err on the side of underestimating the rank, for two reasons. First, a lower estimate of rank often yields a more reasonable (i.e., smaller in norm) computed solution, but does not appreciably increase the size of the residual. Peters and Wilkinson (1970) cite the following example:

$$A = \begin{bmatrix} 6 & 3.0000\ 00000 \\ 4 & 1.9999\ 99998 \\ 2 & 1.0000\ 00003 \end{bmatrix} \qquad b = \begin{bmatrix} 3.0000\ 00000 \\ 2.0004\ 00000 \\ 0.9994\ 00000 \end{bmatrix} .$$

If the matrix A is considered to have rank one, the solution is very close to $(0.4,\ 0.2)^T$, and the residual vector is of order 10^{-4}. On the other hand, if the matrix is considered to have rank two, the solution will be very close to $(10^5, -2 \times 10^5 + 1)$, and the residual is of order 10^{-9}. Although the very large solution vector yields a smaller residual, such a solution may be nonsensical in practice, and, in fact, the first vector is almost certain to be a much "better" solution, despite the larger residual.

A second reason for tending to favor a conservative estimate of the rank is that the perturbation analysis of the least-squares problem shows that the relative change in the exact solution can include a factor $(\text{cond}(A))^2$ for an incompatible right-hand side (see Stewart, 1973). Thus, in order to avoid an ill-conditioned problem, one might wish to accept a smaller value of the rank, and thereby to select a set of columns that are presumed to be "strongly" linearly independent. Although a larger estimate of the rank would be possible with a more stringent criterion for dependence, the conditioning of the least-squares problem might then reflect the square of a much increased condition number for the extended set of columns.

There are, nonetheless, instances where one may wish to allow a very liberal estimate of the rank to be made during the triangular reduction. A notable example occurs for linearly constrained least-squares problems, where the linear constraints are often known to remove the difficulties with near linear dependence in the unconstrained problem. In this case, it would not be appropriate to terminate the triangular reduction prematurely, since the possible dangers from over-estimating the rank would be avoided.

In summary, the best strategy for estimating the rank of a matrix during reduction to triangular form depends on properties of the ultimate problem to be solved.

5. The Complete Orthogonal Factorization

The column interchange strategy described in Section 4.2 allows the procedures of Sections 2.1 and 3.2 to be applied to compute the complete orthogonal factorization of a general real m by n matrix A of rank r. If column interchanges are carried out to ensure that r linearly independent columns are processed first, the configuration after the reduction from the left is:

$$QAP = \begin{bmatrix} \bar{R} \\ \\ 0 \end{bmatrix} ,$$

where $\bar{R}$ is upper trapezoidal, and P is a permutation matrix that specifies the column interchanges. The reduction of $\bar{R}$ to upper triangular form is then carried out if necessary (see Section 3.2), and the orthogonal permutation matrix P is absorbed into the orthogonal matrix arising from the reduction on the right (this simply interchanges the rows of the latter matrix). The final result is

$$QAV = \begin{array}{cc} \overbrace{}^{r} & \overbrace{}^{n-r} \\ \left[\begin{matrix} R & 0 \\ 0 & \end{matrix}\right. & \left.\begin{matrix} \\ \\ \end{matrix}\right] \end{array} \Big\} \, m-r \quad ,$$

and the complete orthogonal factorization of A is given by the matrices Q, V, and R.

This decomposition is extremely useful in many applications. Let Q be partitioned as:

$$Q = \begin{array}{c} \overbrace{\hphantom{Q_1^T}}^{m} \\ \left[\begin{array}{c} Q_1^T \\ \\ Q_2^T \end{array}\right] \begin{array}{l} \}\ r \\ \\ \}\ m - r \end{array} \end{array} .$$

The columns of Q_1 form an orthogonal basis for the space spanned by the columns of A, and the columns of Q_2 form an orthogonal basis for the set of vectors orthogonal to the columns of A. In addition, the last $(n - r)$ columns of V form an orthogonal basis for the set of vectors orthogonal to the <u>rows</u> of A.

Any m-vector y may be written as the sum of two orthogonal parts, which lie respectively in the column space of A and its orthogonal complement, i.e.,

$$y = Q_1 y_1 + Q_2 y_2 .$$

The vectors y_1 and y_2 can be obtained by forming Qy since:

$$Qy = \begin{bmatrix} Q_1^T \\ \\ Q_2^T \end{bmatrix} (Q_1 y_1 + Q_2 y_2) = \begin{bmatrix} y_1 \\ \\ y_2 \end{bmatrix} .$$

Thus, the matrix Q can be applied to compute the projection of a vector into the column space of A or its orthogonal complement, without using projection matrices (see Stewart, 1973, for further details). Since Q is the product of a sequence of Householder transformations, it is not necessary for this purpose to form Q explicitly, but merely to apply the transformations to the given vector.

The bases represented by the columns of Q are also useful in other contexts -- in particular, for solving optimization problems with linear constraints. Suppose that a problem contains the n linearly independent linear constraints

$$A^T y = c \quad , \tag{15}$$

where y is an m-vector. The complete orthogonal factorization of A provides a straightforward way to reduce the dimensionality of the optimization problem. A vector $\bar{y}$ that satisfies (15) can be found by solving the least-squares problem:

$$\min \|A^T y - c\|_2^2 \quad ,$$

which will be compatible if A has full rank. Every vector $\tilde{y}$ satisfying (15) can then be written as:

$$\tilde{y} = \bar{y} + Q_2 v \quad , \tag{16}$$

for some (m - n)-vector v, because the columns of Q_2 are an orthogonal basis for the set of vectors orthogonal to the columns of A. Therefore, only the optimal choice of v is unknown, and any further minimization will occur with respect to these $(m - n)$ variables. Furthermore, every iterate $\tilde{y}$ can be represented in the form (16), and will satisfy the linear constraints by construction of Q_2. These ideas may also be applied in other contexts -- for example, linear inequality constraints (Gill and Murray, 1974), or separable nonlinear least-squares with separable nonlinear equality constraints (Kaufman and Pereyra, 1978).

6. Documentation; Computational Details

There are three major categories of subroutines in this collection, in addition to the general subroutine MNLNLS:

(1) subroutines to carry out the Householder reduction of a given matrix to suitable form. This group includes the subroutines HREDL, HMULTC, LENSQ, and HREDR;

(2) subroutines that solve the least-squares problem after the matrix has been factorized. This group includes the subroutines QRVSLV, QMULVC, VMULVC, TRSLV, TRTSLV, and UNSCRM;

(3) subroutines that explicitly form the orthogonal matrices Q (or Q^T) and V. This group includes the subroutines FRMORT and FORMV.

The subroutines in this package have been designed to include substantial flexibility, to allow their use in widely varying contexts. The user who simply wants to solve the linear least-squares problem, and does not wish to be concerned with any details of the algorithms or fine points of the code, should use the subroutine MNLNLS; MNLNLS is documented in Section 6.7, and automatically calls the appropriate routines. Otherwise, the following brief alphabetical guide indicates the purpose of the subroutines:

-- FORMV (Section 6.1) -- form the explicit orthogonal matrix V;

-- FRMORT (Section 6.2) -- form the explicit orthogonal matrix Q (or Q^T);

-- HMULTC (Section 6.3) -- construct and apply a single Householder transformation to a set of columns;

-- HREDL (Section 6.4) -- carry out the Householder reduction from the left, including estimation of rank;

-- HREDR (Section 6.5) -- carry out the Householder reduction from the right of an upper trapezoidal matrix;

-- LENSQ (Section 6.6) -- compute the squared Euclidean length of a vector;

-- MNLNLS (Section 6.7) -- compute the minimum-length least-squares solution;

-- QMULVC (Section 6.8) -- apply to a vector the sequence of transformations constructed to reduce a matrix from the left;

-- QRVSLV (Section 6.9) -- solve the least-squares problem after the matrix has been reduced to an appropriate form;

-- TRSLV (Section 6.10) -- solve a triangular system of linear equations;

-- TRTSLV (Section 6.11) -- solve the transpose of a triangular system of linear equations;

-- UNSCRM (Section 6.12) -- apply a permutation to a vector;

-- VMULVC (Section 6.13) -- apply to a vector the sequence of transformations constructed to reduce a trapezoidal matrix from the right.

6.1. Subroutine FORMV

6.1.1. Purpose

The subroutine FORMV forms the explicit n by n orthogonal matrix V that is the product of r Householder transformations applied on the right by the subroutine HREDR to reduce an r by n upper trapezoidal matrix $\bar{R}$ to an r by r upper triangle followed by a block of zeros. The first r columns of V form an orthogonal basis for the set of vectors spanned by the rows of $\bar{R}$; the last $(n - r)$ columns of V form an orthogonal basis for the set of vectors orthogonal to the rows of $\bar{R}$. Details are given in Section 3.2.

6.1.2. Description of method

The matrix V is the product of the r transformations applied to $\bar{R}$ on the right:

$$V = \bar{H}_r \bar{H}_{r-1} \cdots \bar{H}_1 ,$$

where the vector corresponding to $\bar{H}_j$ has non-zeros in components j, and $(r + 1)$ through n. To construct V, these transformations are multiplied together beginning at the right, after $\bar{H}_1$ is formed explicitly. Finally, any column interchanges carried out during the reduction from the left to upper trapezoidal form are incorporated by applying the appropriate permutation matrix on the left (interchanging the rows of the orthogonal matrix).

6.1.3. Keywords

Householder reduction from right; complete orthogonal factorization.

6.1.4. Source language

Fortran. The code in FORMV has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.1.5. Specification and parameters

See accompanying listing.

6.1.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.1.7. Auxiliary routines

FORMV calls the standard function DABS.

6.1.8. Program size

71 Fortran source statements.

6.1.9. Array storage

No locally declared arrays.

6.1.10. Timing

The number of arithmetic operations required to form V is of approximate order $r^2(n - r) + 2r(n - r)^2$.

6.1.11. Further Comments

None.

```
      SUBROUTINE FORMV ( NCOL,  NDIM,  QRV,  NRANK,  HVECR,  IPERM,
     1     NVDIM,  V,  TEMP,  LERROR )
C
      INTEGER    NCOL, NDIM, NRANK, NVDIM, LERROR
      INTEGER    IPERM(NCOL)
      DOUBLE PRECISION   QRV(NDIM, NCOL), HVECR(NCOL), V(NVDIM, NCOL),
     1   TEMP(NCOL)
C
C
C-----------------------------------------------------------------------
C
C
C        THE SUBROUTINE FORMV IS USED IN CONJUNCTION WITH THE SUBROUTINES
C     HREDL AND HREDR TO COMPUTE PART OF THE COMPLETE ORTHOGONAL
C     FACTORIZATION OF A GENERAL REAL MATRIX.
C
C        FOR EVERY REAL NROW BY NCOL REAL MATRIX QRV OF RANK NRANK,
C     THERE EXIST AN NROW BY NROW ORTHOGONAL MATRIX Q, AN NCOL BY NCOL
C     ORTHOGONAL MATRIX V, AND AN NRANK BY NRANK UPPER TRIANGULAR MATRIX
C     R, SUCH THAT
C
C        Q * QRV * V = < R    0 >
C                      <    0   > .
C
C        THE MATRIX Q IS THE PRODUCT OF NRANK HOUSEHOLDER TRANSFORMATIONS
C     CONSTRUCTED BY THE SUBROUTINE HREDL.  IF NRANK .EQ. NCOL, THE MATRIX
C     V IS SIMPLY THE PERMUTATION MATRIX DEFINED BY ANY COLUMN INTER-
C     CHANGES MADE DURING EXECUTION OF HREDL.  IF NRANK .LT. NCOL, V IS
C     THE PRODUCT OF THE PERMUTATION MATRIX AND A SEQUENCE OF NRANK
C     HOUSEHOLDER TRANSFORMATIONS CONSTRUCTED DURING EXECUTION OF THE
C     SUBROUTINE HREDR.  THE LAST (NCOL-NRANK) COLUMNS OF V FORM AN
C     ORTHOGONAL BASIS FOR THE SUBSPACE OF VECTORS ORTHOGONAL TO THE
C     ROWS OF THE ORIGINAL MATRIX QRV.
C
C
C        THE SEQUENCE OF HOUSEHOLDER TRANSFORMATIONS APPLIED ON THE
C     RIGHT BY HREDR MAY BE WRITTEN AS
C
C           P(NRANK) * P(NRANK-1) * ... * P(2) * P(1) ,
C
C     WHERE EACH P(J) IS A HOUSEHOLDER MATRIX CHOSEN TO REDUCE ROW J TO
C     THE APPROPRIATE FORM.  THE VECTOR U(J) CORRESPONDING TO P(J) IS OF
C     THE FOLLOWING SPECIAL FORM:  COMPONENTS 1 THROUGH (J-1) ARE ZERO,
C     COMPONENT J IS NON-ZERO, COMPONENTS (J+1) THROUGH NRANK ARE ZERO,
C     AND COMPONENTS NRANK+1 THROUGH NCOL ARE NON-ZERO.  DURING EXECUTION
C     OF FORMV, THESE TRANSFORMATIONS ARE MULTIPLIED TOGETHER FROM RIGHT
C     TO LEFT, TAKING ADVANTAGE OF THE SPECIAL STRUCTURE OF THE U(J) TO
C     SAVE ARITHMETIC OPERATIONS.  THEN, THE PERMUTATION MATRIX THAT
C     DEFINES ANY COLUMN INTERCHANGES MADE DURING THE REDUCTION FROM THE
C     LEFT IS APPLIED TO YIELD THE DESIRED MATRIX, V.
```

```
C
C       FORMV CALLS THE SUBROUTINE UNSCRM.
C
C
C  THE FORMAL PARAMETERS OF FORMV ARE :
C
C  NCOL --
C       INTEGER, INPUT ONLY.
C       THE NUMBER OF COLUMNS OF THE ORIGINAL MATRIX, QRV.
C
C  NDIM --
C       INTEGER, INPUT ONLY.
C       THE DECLARED ROW DIMENSION OF THE MATRIX QRV IN THE CALLING
C       SUB-PROGRAM.
C
C  QRV --
C       DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NDIM BY NCOL,
C       INPUT ONLY.
C       THE MATRIX QRV CORRESPONDS TO THE ORIGINAL MATRIX THAT WAS
C       REDUCED BY HREDL AND HREDR.  ITS CONTENTS MUST NOT BE ALTERED
C       AFTER EXIT FROM HREDR AND BEFORE ENTRY TO FORMV.
C
C  NRANK --
C       INTEGER, INPUT ONLY.
C       THE NUMERICAL RANK OF THE ORIGINAL MATRIX QRV, AS DETERMINED BY
C       HREDL.  THE VALUE OF NRANK MUST NOT BE ALTERED AFTER EXIT FROM
C       HREDL AND BEFORE ENTRY TO FORMV.
C
C  HVECR --
C       DOUBLE PRECISION VECTOR, OF LENGTH NCOL, INPUT ONLY.
C       THE VECTOR HVECR IS GENERATED BY HREDR, AND ITS CONTENTS MUST
C       NOT BE ALTERED AFTER EXIT FROM HREDR AND BEFORE ENTRY TO FORMV.
C
C  IPERM --
C       INTEGER VECTOR, OF LENGTH NCOL, INPUT ONLY.
C       THE VECTOR IPERM IS GENERATED BY HREDL, AND ITS CONTENTS MUST
C       NOT BE ALTERED AFTER EXIT FROM HREDL AND BEFORE ENTRY TO FORMV.
C
C  NVDIM --
C       INTEGER, INPUT ONLY.
C       THE DECLARED ROW DIMENSION OF THE MATRIX V IN THE CALLING
C       SUB-PROGRAM.  MUST BE .GE. NCOL.
C
C  V --
C       DOUBLE PRECISION ARRAY, OF CONCEPTUAL DIMENSION NCOL BY NCOL,
C       AND DECLARED DIMENSION NVDIM BY NCOL.  OUTPUT ONLY.  ON EXIT
C       FROM FORMV, THE V MATRIX CONTAINS THE DESIRED ORTHOGONAL
C       MATRIX, V.
C
C  TEMP --
```

```
C         DOUBLE PRECISION VECTOR, OF LENGTH NCOL, OUTPUT ONLY.
C         USED FOR INTERMEDIATE STORAGE DURING EXECUTION OF FORMV.
C
C   LERROR --
C         INTEGER, OUTPUT ONLY.
C         AN ERROR INDICATOR, WITH THE FOLLOWING POSSIBLE VALUES:
C         LERROR = 0 : NO ERRORS, NORMAL TERMINATION.
C         LERROR = 1 : INVALID INPUT PARAMETER.
C
C
C
C    *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                  SYSTEMS OPTIMIZATION LABORATORY
C                  DEPARTMENT OF OPERATIONS RESEARCH
C                  STANFORD UNIVERSITY
C                  STANFORD, CALIFORNIA 94305
C
C    *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C       DECLARATION OF LOCAL VARIABLES.
C
        INTEGER   I, J, JP1, K, NCOLM1, NRNKP1
        DOUBLE PRECISION   BETA, DOT, PRDJ, PROD
C
C       DECLARATION OF STANDARD FUNCTIONS.
C
        DOUBLE PRECISION   DABS
C
C-----------------------------------------------------------------------
C
C   TEST FOR INVALID VALUES OF NRANK AND NCOL.
C
C-----------------------------------------------------------------------
C
        LERROR = 1
        IF (NCOL .LE. 0 .OR. NRANK .LE. 0 .OR. NRANK .GT. NCOL)  RETURN
        DO 20 J = 1, NCOL
           DO 10 I = 1, NCOL
              V(I,J) = 0.0D+0
   10      CONTINUE
           V(J,J) = 1.0D+0
   20   CONTINUE
        LERROR = 0
        IF (NRANK .EQ. NCOL)  GO TO 210
        NRNKP1 = NRANK + 1
C
```

```
C---------------------------------------------------------------------
C
C  FORM THE RIGHT-MOST HOUSEHOLDER MATRIX EXPLICITLY.
C
C---------------------------------------------------------------------
C
      IF (HVECR(1) .EQ. 0.0D+0)  GO TO 70
      BETA = 1.0D+0/DABS(HVECR(1))
      PROD = 1.0D+0
      IF (HVECR(1) .LT. 0.0D+0)  PROD = -1.0D+0
      V(1,1) = 1.0D+0 - PROD*HVECR(1)
      DO 30 J = NRNKP1, NCOL
         V(1,J) = -PROD*QRV(1,J)
         V(J,1) = V(1,J)
   30 CONTINUE
      IF (NRNKP1 .EQ. NCOL)  GO TO 60
      NCOLM1 = NCOL - 1
      DO 50 J = NRNKP1, NCOLM1
         PRDJ = BETA*QRV(1,J)
         V(J,J) = 1.0D+0 - PRDJ*QRV(1,J)
         JP1 = J + 1
         DO 40 I = JP1, NCOL
            V(I,J) = -PRDJ*QRV(1,I)
            V(J,I) = V(I,J)
   40    CONTINUE
   50 CONTINUE
C
C     FILL IN THE (NCOL, NCOL) ELEMENT.
C
   60 CONTINUE
      V(NCOL,NCOL) = 1.0D+0 - BETA*QRV(1,NCOL)**2
   70 IF (NRANK .EQ. 1)  RETURN
C
C---------------------------------------------------------------------
C
C  MULTIPLY TOGETHER THE REMAINING (NRANK-1) TRANSFORMATIONS,
C  APPLYING EACH NEW TRANSFORMATION FROM THE LEFT.
C
C---------------------------------------------------------------------
C
      DO 200 K = 2, NRANK
C
C        BETA IS THE NORMALIZING FACTOR FOR THE K-TH TRANSFORMATION.
C
         IF (HVECR(K) .EQ. 0.0D+0)  GO TO 200
         BETA = 1.0D+0/DABS(HVECR(K))
C
C        ONLY ROW K AND ROWS NRANK+1 THROUGH NCOL OF THE CURRENT V
C        ARE ALTERED BY APPLICATION OF THE K-TH TRANSFORMATION ON THE
C        LEFT.
```

```
C
C
C         COLUMN K OF THE ALTERED V SIMPLY BECOMES THE K-TH COLUMN OF
C         THE K-TH HOUSEHOLDER TRANSFORMATION.
C
          PROD = 1.0D+0
          IF (HVECR(K) .LT. 0.0D+0)  PROD = -1.0D+0
          V(K,K) = 1.0D+0 - PROD*HVECR(K)
          DO 100 I = NRNKP1, NCOL
             V(I,K) = -PROD*QRV(K,I)
  100     CONTINUE
C
C
C         ALTER COLUMNS 1 THROUGH K-1 AND NRANK + 1 THROUGH NCOL,
C         USING A SINGLE DO LOOP FOR COMPACTNESS.
C
C         THE LOOP TO STATEMENT 130 IS A LOOP OVER THE COLUMNS OF THE
C         CURRENT V.
C
          DO 130 J = 1, NCOL
C
C            DO NOTHING IF J .GE. K AND J .LE. NRANK.  THESE
C            COLUMNS ARE UNALTERED BECAUSE THEIR INNER PRODUCT WITH
C            THE K-TH HOUSEHOLDER VECTOR IS ZERO.
C
             IF (J .GE. K .AND. J .LE. NRANK)  GO TO 130
C
C
C            FORM THE INNER PRODUCT OF THE K-TH HOUSEHOLDER VECTOR
C            AND THE J-TH COLUMN OF THE CURRENT V.
C
             DOT = 0.0D+0
             DO 110 I = NRNKP1, NCOL
                DOT = DOT + QRV(K,I)*V(I,J)
  110        CONTINUE
             PROD = BETA*DOT
C
C            ALTER ROW K BY SUBTRACTING A MULTIPLE OF THE K-TH ELEMENT OF
C            THE HOUSEHOLDER VECTOR ( STORED IN HVECR(K) ).
C
             V(K,J) = V(K,J) - PROD*HVECR(K)
C
C            ALTER ROWS NRANK+1 THROUGH NCOL OF COLUMN J.
C
             DO 120 I = NRNKP1, NCOL
                V(I,J) = V(I,J) - PROD*QRV(K,I)
  120        CONTINUE
C
C         END LOOP OVER THE COLUMNS OF V.
C
```

```
  130     CONTINUE
C
C       END LOOP OVER THE HOUSEHOLDER TRANSFORMATIONS.
C
  200 CONTINUE
C
C
C-----------------------------------------------------------------------
C
C  PERMUTE THE ROWS OF V, ACCORDING TO THE INTERCHANGES CARRIED OUT ON
C  THE COLUMNS OF QRV DURING ITS REDUCTION FROM THE LEFT.
C
C-----------------------------------------------------------------------
C
  210 DO 230 J = 1, NCOL
          CALL UNSCRM( 1, NCOL, IPERM, V(1,J), TEMP )
          DO 220 I = 1, NCOL
             V(I,J) = TEMP(I)
  220     CONTINUE
  230 CONTINUE
      RETURN
      END
```

6.2. Subroutine FRMORT

6.2.1. Purpose

The subroutine FRMORT forms the explicit orthogonal matrix Q that is the product of r Householder transformations applied on the left by the subroutine HREDL to reduce an m by n matrix of rank r to upper trapezoidal form; alternatively, the matrix Q^T may be formed.

The first r rows of Q (columns of Q^T) form an orthogonal basis for the subspace of vectors spanned by the columns of the original matrix, and the last $(n - r)$ rows of Q (columns of Q^T) form an orthogonal basis for the set of vectors orthogonal to these columns.

6.2.2. Description of method

The matrix Q is the product of the r transformations as applied during the reduction from the left:

$$Q = H_r H_{r-1} \cdots H_2 H_1 \quad ,$$

where the vector corresponding to H_j is zero in components 1 through $j - 1$. The matrix Q^T is the product of these transformations in reverse order:

$$Q^T = H_1 H_2 \cdots H_{r-1} H_r \quad .$$

To form Q or Q^T, the transformations are multiplied together, starting with H_r in explicit form -- for example, in forming Q each successive transformation is applied on the right, leading to the recursive definition:

$$Q_r = H_r \quad ;$$

$$\text{for } k = r - 1 \text{ step } -1 \text{ until } 1:$$

$$Q_k = Q_{k+1} H_k \quad .$$

Then:

$$Q \equiv Q_1 \quad .$$

In applying H_k, full advantage is taken of its special structure, and the known form of the partial product to which it is applied (e.g., Q_k contains a $(k - 1)$ by $(k - 1)$ identity matrix in its upper left corner). Complete details are given through comments in the code.

6.2.3. Keywords

Complete orthogonal factorization; QR factorization.

6.2.4. Source language

Fortran. The code in FRMORT has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.2.5. Specification and parameters

See accompanying listing.

6.2.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.2.7. Auxiliary routines

FRMORT calls the standard functions DABS and IABS.

6.2.8. Program size

89 Fortran source statements.

6.2.9. Array storage

No locally declared arrays.

6.2.10. Timing

The number of arithmetic operations required to form Q is of approximate order $2mr(m - r) + \frac{2}{3} r^3$.

6.2.11. Further Comments

None.

```
      SUBROUTINE FRMORT ( NROW,   NRNK,   NDIM,   QRV,   HVECL,   MULORD,
     1                   NODIM,   ORTH,   IERROR )
C
      INTEGER    NROW, NRNK, NDIM, MULORD, NODIM, IERROR
      DOUBLE PRECISION   QRV(NDIM, NRNK), HVECL(NRNK), ORTH(NODIM, NROW)
C
C
C-----------------------------------------------------------------------
C
C        THE SUBROUTINE FRMORT IS USED (IN CONJUNCTION WITH THE
C     SUBROUTINE HREDL) TO FORM THE EXPLICIT ORTHOGONAL MATRIX THAT IS
C     THE PRODUCT OF A SPECIAL SEQUENCE OF HOUSEHOLDER TRANSFORMATIONS,
C     MULTIPLIED IN EITHER FORWARD OR BACKWARD ORDER.  THE SEQUENCE OF
C     TRANSFORMATIONS ARISES WHEN THE SUBROUTINE HREDL IS USED TO REDUCE
C     THE MATRIX QRV TO UPPER TRAPEZOIDAL FORM.  THIS REDUCTION INVOLVES
C     A SEQUENCE OF HOUSEHOLDER TRANSFORMATIONS APPLIED ON THE LEFT TO
C     QRV (POSSIBLY WITH ITS COLUMNS INTERCHANGED), AND MAY BE WRITTEN
C
C           P(NRNK) * ... * P(1) * (PERMUTED QRV) = (UPPER TRAPEZOID).
C
C        EACH P(J) IS A HOUSEHOLDER MATRIX, OF THE FORM
C
C           I - (U(J) * U(J) TRANSPOSE) / BETA(J),
C
C     WHERE U(J) IS A VECTOR WHOSE FIRST (J-1) COMPONENTS ARE ZERO, AND
C     BETA(J) IS A SCALAR NORMALIZING FACTOR.  THE SUBROUTINE HREDL
C     CARRIES OUT THIS REDUCTION, AND STORES INFORMATION ABOUT THE
C     TRANSFORMATIONS IN COMPACT FORM IN THE MATRIX QRV AND IN SEVERAL
C     AUXILIARY VECTORS (SEE THE DOCUMENTATION OF HREDL FOR DETAILS).
C
C        THE SUBROUTINE FRMORT USES THE OUTPUT OF HREDL TO GENERATE
C     EITHER THE MATRIX Q DEFINED AS THE PRODUCT OF THE TRANSFORMATIONS
C     IN FORWARD ORDER, I.E.,
C
C                 Q = P(NRNK) * ... * P(2) * P(1),
C
C     OR THE TRANSPOSE OF Q  (QT), WHICH IS GIVEN BY THE PRODUCT OF THE
C     TRANSFORMATIONS IN REVERSE ORDER, I.E.,
C
C                 QT = P(1) * P(2) * ... * P(NRNK).
C
C
C        THE ROWS OF Q (COLUMNS OF QT) PROVIDE USEFUL INFORMATION
C     ABOUT THE COLUMN SPACE OF THE ORIGINAL MATRIX.  LET 'NRNK' BE THE
C     ESTIMATED NUMERICAL RANK OF THE ORIGINAL MATRIX, AS DETERMINED
C     DURING EXECUTION OF HREDL.  THE FIRST NRNK ROWS OF Q (COLUMNS OF
C     QT) FORM AN ORTHOGONAL BASIS FOR THE SPACE SPANNED BY THE COLUMNS
C     OF THE ORIGINAL MATRIX, AND THE REMAINING ROWS OF Q (COLUMNS OF
C     QT) FORM AN ORTHOGONAL BASIS FOR THE SPACE OF VECTORS ORTHOGONAL
C     TO THE COLUMNS OF THE ORIGINAL MATRIX.
```

```
C
C       ** CAUTION ** ADVANTAGE IS TAKEN OF THE SPECIAL FORM OF THE
C  TRANSFORMATIONS IN FORMING THE PRODUCT, SO THAT FRMORT IS NOT
C  SUITABLE FOR MULTIPLYING A GENERAL SEQUENCE OF HOUSEHOLDER
C  TRANSFORMATIONS.
C
C
C  NROW --
C          INTEGER, INPUT ONLY.
C          THE NUMBER OF ROWS OF THE ORIGINAL MATRIX QRV THAT WAS
C          REDUCED BY THE SUBROUTINE HREDL.  MUST BE .GT. 0.
C
C  NRNK --
C          INTEGER, INPUT ONLY.
C          THE NUMBER OF TRANSFORMATIONS APPLIED BY HREDL (THE
C          ESTIMATED RANK).  THE VALUE OF NRNK MUST NOT BE ALTERED
C          AFTER EXIT FROM HREDL AND BEFORE ENTRY TO FRMORT.
C
C  NDIM --
C          INTEGER, INPUT ONLY.
C          THE DECLARED ROW DIMENSION OF THE MATRIX QRV IN THE CALLING
C          SUB-PROGRAM.  MUST BE .GE. NROW.
C
C  QRV --
C          DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NDIM BY NRNK,
C          AND CONCEPTUAL DIMENSION NROW BY NRNK.  INPUT ONLY.
C          THE MATRIX QRV CORRESPONDS TO THE ORIGINAL MATRIX THAT WAS
C          REDUCED BY HREDL.  ITS CONTENTS MUST NOT BE ALTERED AFTER
C          EXIT FROM HREDL AND BEFORE ENTRY TO FRMORT.  A DESCRIPTION
C          OF THE CONTENTS OF QRV IS GIVEN IN THE DOCUMENTATION FOR
C          HREDL.
C
C  HVECL --
C          DOUBLE PRECISION VECTOR OF LENGTH NRNK, INPUT ONLY.
C          THE VECTOR HVECL IS GENERATED BY HREDL, AND ITS CONTENTS MUST
C          NOT BE ALTERED AFTER EXIT FROM HREDL AND BEFORE ENTRY TO
C          FRMORT.  A DESCRIPTION OF THE CONTENTS OF HVECL IS GIVEN
C          IN THE DOCUMENTATION FOR HREDL.
C
C  MULORD --
C          INTEGER, INPUT ONLY.
C          THE SIGN OF MULORD INDICATES THE ORDER IN WHICH THE TRANS-
C          FORMATIONS ARE TO BE MULTIPLIED.  IF MULORD = +1, THEY ARE
C          MULTIPLIED IN FORWARD ORDER, TO PRODUCE Q.  IF MULORD = -1,
C          THEY ARE MULTIPLIED IN REVERSE ORDER, TO PRODUCE Q TRANSPOSE.
C
C  NODIM --
C          INTEGER, INPUT ONLY.
C          THE DECLARED ROW DIMENSION OF THE MATRIX QRTH IN THE CALLING
C          SUB-PROGRAM.  MUST BE .GE. NROW.
```

```
C
C  ORTH --
C         DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NODIM BY NROW,
C         AND CONCEPTUAL DIMENSION NROW BY NROW.  OUTPUT ONLY.
C         ON EXIT FROM FRMORT, THE MATRIX ORTH CONTAINS EITHER Q OR QT
C         (Q TRANSPOSE), DEPENDING ON THE SIGN OF MULORD.
C
C  LERROR --
C         INTEGER, OUTPUT ONLY.
C         ERROR INDICATOR, SET DURING EXECUTION OF FRMORT, WITH THE
C         FOLLOWING POSSIBLE VALUES.
C            LERROR = 0 : NO ERRORS, NORMAL TERMINATION.
C            LERROR = 1 : INVALID INPUT PARAMETER.
C            LERROR = 2 : NRNK IS ZERO.  IF LERROR=2, Q IS SET TO
C                         THE IDENTITY MATRIX.
C
C
C
C   *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                 SYSTEMS OPTIMIZATION LABORATORY
C                 DEPARTMENT OF OPERATIONS RESEARCH
C                 STANFORD UNIVERSITY
C                 STANFORD, CALIFORNIA 94305
C
C   *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C      DECLARATION OF LOCAL VARIABLES.
C
       INTEGER   I, J, JP1, K, KK, KP1, NRNKP1, NRNKM1, NROWM1
       DOUBLE PRECISION   BETA, DOTPRD, FACTOR, FCTIND, FCTJ, FCTRNK
C
C      DECLARATION OF STANDARD FUNCTIONS.
C
       INTEGER   IABS
       DOUBLE PRECISION   DABS
C
C
C  TEST FOR ERROR IN INPUT PARAMETERS.
C
       LERROR = 1
       IF (NROW .LE. 0 .OR. NRNK .LT. 0 .OR. NDIM .LT. NROW .OR.
     1    NODIM .LE. 0 .OR. NRNK .GT. NROW .OR.
     2    IABS(MULORD) .NE. 1)  RETURN
C
C-----------------------------------------------------------------------
C
C  INITIALIZE ORTH TO THE NROW BY NROW IDENTITY MATRIX.
```

```
C
C-----------------------------------------------------------------------
C
      DO 20 J = 1, NROW
         DO 10 I = 1, NROW
            ORTH(I,J)  = 0.0D+0
   10    CONTINUE
         ORTH(J,J)  =  1.0D+0
   20 CONTINUE
      IERROR = 2
      IF (NRNK .EQ. 0)   RETURN
C
C-----------------------------------------------------------------------
C
C  GENERATE P(NRNK) EXPLICITLY, SINCE ITS FORM IS THAT OF A SINGLE
C  HOUSEHOLDER TRANSFORMATION.
C
C-----------------------------------------------------------------------
C
      IF (HVECL(NRNK) .EQ. 0.0D+0)  GO TO 80
      BETA = DABS(HVECL(NRNK))
      FACTOR  =  1.0D+0/BETA
      ORTH(NRNK,NRNK)  =  1.0D+0 - BETA
      IF (NRNK .EQ. NROW)   GO TO 70
C
C-----------------------------------------------------------------------
C
C  FILL IN THE LOWER RIGHT-HAND (NROW-NRNK) BY (NROW-NRNK) SQUARE OF
C  P(NRNK); THE REMAINDER OF P(NRNK) IS SIMPLY THE IDENTITY.
C
C  THE NRNK-TH ROW AND COLUMN OF P(NRNK) ARE FORMED SEPARATELY
C  SINCE THE NRNK-TH COMPONENT OF U(NRNK) IS STORED IN HVECL(NRNK).
C  THE REMAINING COMPONENTS OF U(NRNK) ARE STORED BELOW THE DIAGONAL
C  IN THE NRNK-TH COLUMN OF QRV.
C
C-----------------------------------------------------------------------
C
      NRNKP1 = NRNK + 1
      FCTRNK  =  FACTOR*HVECL(NRNK)
      DO 30 I  = NRNKP1, NROW
         ORTH(I,NRNK)  =  -FCTRNK*QRV(I,NRNK)
         ORTH(NRNK,I)  =  ORTH(I,NRNK)
   30 CONTINUE
      IF (NRNKP1 .GE. NROW)   GO TO 60
      NROWM1 = NROW - 1
C
C     THE LOOP TO STATEMENT 50 COMPUTES ALL ELEMENTS OF THE SUB-MATRIX
C     EXCEPT THE NROW-TH DIAGONAL ELEMENT.
C
      DO 50 J = NRNKP1, NROWM1
```

```
            JP1 = J + 1
            FCTJ = FACTOR*QRV(J,NRNK)
            DO 40 I = JP1, NROW
               ORTH(I,J) = -FCTJ*QRV(I,NRNK)
               ORTH(J,I) = ORTH(I,J)
   40       CONTINUE
            ORTH(J,J) = 1.0D+0 - FACTOR*QRV(J,NRNK)**2
   50 CONTINUE
C
C     DEFINE THE NROW-TH DIAGONAL ELEMENT
C
   60 ORTH(NROW,NROW) = 1.0D+0 - FACTOR*QRV(NROW,NRNK)**2
   70 CONTINUE
   80 LERROR = 0
      IF (NRNK .EQ. 1)  RETURN
C
C-----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 200 MULIPLIES TOGETHER THE REMAINING
C  HOUSEHOLDER TRANSFORMATIONS.  BEFORE BEGINNING THE LOOP, ORTH
C  CONTAINS P(NRNK).  EACH TIME THROUGH THE LOOP, THE CURRENT
C  ORTHOGONAL MATRIX IS MULTIPLIED EITHER ON THE RIGHT (IF MULORD = +1)
C  OR ON THE LEFT (IF MULORD = -1) BY A HOUSEHOLDER TRANSFORMATION
C  CORRESPONDING TO A VECTOR WITH ONE MORE NON-ZERO COMPONENT.
C
C-----------------------------------------------------------------------
C
      NRNKM1 = NRNK - 1
      DO 200 KK = 1, NRNKM1
C
C        FOR K = NRNK-1 STEP -1 UNTIL 1
C
         K = NRNKM1 - KK + 1
C
C        TEST WHETHER THE K-TH TRANSFORMATION WAS SKIPPED.
C
         IF (HVECL(K) .EQ. 0.0D+0)  GO TO 200
         BETA = DABS(HVECL(K))
         FACTOR = 1.0D+0/BETA
         IF (MULORD .LT. 0)  GO TO 150
C
C        ---------------------------------------------------------------
C
C        THE K-TH TRANSFORMATION IS TO BE APPLIED ON THE RIGHT.
C
C        THE FIRST (K-1) ROWS OF THE CURRENT Q ARE UNALTERED
C        WHEN P(K) IS APPLIED FROM THE RIGHT, SINCE
C        Q(I)**T*U(K) = 0, I = 1,...,K-1, WHERE Q(I)**T IS THE
C        I-TH ROW OF THE CURRENT Q.  THE FIRST (K-1) COLUMNS OF THE
C        EXISTING Q ARE ALS  UNALTERED, SINCE THE FIRST (K-1)
```

```
C         COMPONENTS OF U(K) ARE ZERO, AND THE FIRST K ROWS OF
C         THE CURRENT Q ARE ROWS OF THE IDENTITY MATRIX.
C
C         COMPUTE THE K-TH ROW OF THE MODIFIED Q, WHICH IS SIMPLY
C         THE K-TH ROW OF P(K),  BECAUSE THE K-TH ROW OF
C         THE UNMODIFIED Q IS THE K-TH ROW OF THE IDENTITY.
C
C         ---------------------------------------------------------------
C
          ORTH(K,K)  =  1.0D+0 - FACTOR*HVECL(K)**2
          KP1 = K + 1
          FCTIND = FACTOR*HVECL(K)
          DO 110 J = KP1, NROW
             ORTH(K,J)  =  -FCTIND*QRV(J,K)
  110     CONTINUE
C
C         THE LOOP TO STATEMENT 140 ALTERS ROWS K+1 THROUGH NROW OF
C         THE PREVIOUSLY COMPUTED Q.
C
          DO 140 I = KP1, NROW
C
C            FORM THE SCALAR PRODUCT OF THE I-TH ROW OF THE UNMODIFIED
C            Q AND THE VECTOR U(K).  SINCE THE K-TH COLUMN OF THE
C            UNMODIFIED Q IS ZERO IN ROWS K+1 THROUGH NROW, THE TERM
C            OF THE INNER PRODUCT CORRESPONDING TO THIS COMPONENT OF
C            U(K) IS OMITTED.
C
             DOTPRD = 0.0D+0
             DO 120 J = KP1, NROW
                DOTPRD = DOTPRD + ORTH(I,J)*QRV(J,K)
  120        CONTINUE
C
C            MODIFY THE I-TH ROW BY SUBTRACTING AN APPROPRIATE MULTIPLE
C            OF THE HOUSEHOLDER VECTOR, U(K).
C
             FCTJ = FACTOR*DOTPRD
             ORTH(I,K)  =  ORTH(I,K)  - FCTJ*HVECL(K)
             DO 130 J = KP1, NROW
                ORTH(I,J)  =  ORTH(I,J)  - FCTJ*QRV(J,K)
  130        CONTINUE
C
C         STATEMENT 140 ENDS THE LOOP OVER THE ALTERED ROWS OF Q.
C
  140     CONTINUE
          GO TO 200
C
  150     CONTINUE
C
C         ---------------------------------------------------------------
C
```

```
C         THE K-TH TRANSFORMATION IS TO BE APPLIED ON THE LEFT.
C
C         THE FIRST (K-1) COLUMNS OF THE CURRENT QT ARE UNALTERED
C         WHEN P(K) IS APPLIED FROM THE LEFT, SINCE
C         QT(I)**T*U(K) = 0, I = 1,...,K-1, WHERE QT(I) IS THE
C         I-TH COLUMN OF THE CURRENT QT.  THE FIRST (K-1) ROWS OF THE
C         EXISTING QT ARE ALSO UNALTERED, SINCE THE FIRST (K-1)
C         COMPONENTS OF U(K) ARE ZERO, AND THE FIRST K COLUMNS OF
C         THE CURRENT QT ARE COLUMNS OF THE IDENTITY MATRIX.
C
C         COMPUTE THE K-TH COLUMN OF THE MODIFIED QT, WHICH IS SIMPLY
C         THE K-TH COLUMN OF P(K),  BECAUSE THE K-TH COLUMN OF
C         THE UNMODIFIED QT IS THE K-TH COLUMN OF THE IDENTITY.
C
C         ------------------------------------------------------------
C
          ORTH(K,K)  =  1.0D+0  -  FACTOR*HVECL(K)**2
          KP1 = K + 1
          FCTIND = FACTOR*HVECL(K)
          DO 160 J = KP1, NROW
             ORTH(J,K)  =  -FCTIND*QRV(J,K)
  160     CONTINUE
C
C         THE LOOP TO STATEMENT 190 ALTERS COLUMNS K+1 THROUGH NROW OF
C         THE PREVIOUSLY COMPUTED QT.
C
          DO 190 J = KP1, NROW
C
C            FORM THE SCALAR PRODUCT OF THE J-TH COLUMN OF THE UNMODIFIED
C            QT AND THE VECTOR U(K).  SINCE THE K-TH ROW OF THE
C            UNMODIFIED QT IS ZERO IN COLUMNS K+1 THROUGH NROW, THE TERM
C            OF THE INNER PRODUCT CORRESPONDING TO THIS COMPONENT OF
C            U(K) IS OMITTED.
C
             DOTPRD = 0.0D+0
             DO 170 I = KP1, NROW
                DOTPRD = DOTPRD + ORTH(I,J)*QRV(I,K)
  170        CONTINUE
C
C            MODIFY THE J-TH COLUMN BY SUBTRACTING AN APPROPRIATE
C            MULTIPLE OF THE HOUSEHOLDER VECTOR, U(K).
C
             FCTJ = FACTOR*DOTPRD
             ORTH(K,J)  =  ORTH(K,J)  -  FCTJ*HVECL(K)
             DO 180 I = KP1, NROW
                ORTH(I,J) = ORTH(I,J)  - FCTJ*QRV(I,K)
  180        CONTINUE
C
C         STATEMENT 190 ENDS THE LOOP OVER THE ALTERED COLUMNS OF QT.
C
```

```
  190     CONTINUE
C
C-----------------------------------------------------------------------
C
C     STATEMENT 200 ENDS THE LOOP OVER THE HOUSEHOLDER TRANSFORMATIONS.
C
C-----------------------------------------------------------------------
C
  200 CONTINUE
      RETURN
      END
```

6.3. Subroutine HMULTC

6.3.1. Purpose

The subroutine HMULTC (for Householder transformation multiplied over columns) constructs a single Householder transformation to annihilate a block of components in a particular column of a given matrix, and applies the transformation to the remaining columns. The process of defining the transformation is described in Section 2.1.

6.3.2. Description of method

See Section 2.1.

6.3.3. Keywords

Orthogonal transformation; Householder transformation.

6.3.4. Source language

Fortran. The code in HMULTC has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.3.5. Specification and parameters

See accompanying listing.

6.3.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.3.7. Auxiliary routines

HMULTC requires the standard functions DABS and DSQRT.

6.3.8. Program size

37 Fortran source statements.

6.3.9. Array storage

No locally declared arrays.

6.3.10. Timing

The number of arithmetic operations necessary to construct a single normalized Householder transformation to annihilate k components of an m-vector, and then transform ℓ other m-vectors includes $k + \ell$ divisions, and of order $2\ell(k + 1)$ multiplications/additions.

6.3.11. Further comments

Representation of Householder transformation. A Householder matrix is defined in terms of the non-zero vector u, by:

$$H(u) = I - \frac{2uu^T}{\|u\|^2} \equiv I - \frac{uu^T}{\beta} ,$$

where β (the scaling factor for the transformation) $= 1/2\ \|u\|^2$. To represent such a transformation, it might appear that only the vector

u needs to be retained; however, it would usually be considered wasteful to recompute $1/2\,\|u\|^2$ several times, and thus one might wish to store the quantity β as well.

The transformations used in reducing a matrix to upper trapezoidal form from the left have a special structure (see Section 2.1); in particular, each transformation may be regarded as reducing the "first" column of a remaining matrix. Consequently, we need only consider representing the Householder transformation that reduces a single vector y with ℓ components to a multiple of e_1, i.e., that annihilates all but the first component of y:

$$\left(I - \frac{uu^T}{\beta}\right)y = \rho e_1 = \begin{bmatrix} \rho \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} . \qquad (17)$$

Because Euclidean length is preserved by a Householder transformation, $\rho^2 = \|y\|_2^2$. Multiplying out the first expression in (17), it can be seen that the Householder vector u is a multiple of $(y-\rho e_1)$, i.e.,

$$u = \frac{1}{\gamma} \begin{bmatrix} y_1 - \rho \\ y_2 \\ \vdots \\ y_\ell \end{bmatrix} , \tag{18}$$

for some scalar γ. To avoid cancellation in computing u, the sign of ρ is always chosen to be opposite to that of y_1, so that $\rho = -\text{sign}\,(y_1)\|y\|$, where $\text{sign}(y_1) = 1$ if $y_1 \geq 0$, and $= -1$ otherwise.

The Householder vector u can be stored very efficiently. First, since components 2 through ℓ of y will be annhilated by the transformation, components 2 through ℓ of the Householder vector can be stored in these components of y; hence, only one additional location is required to store the first component of u. Second, the scalar γ in (18) can be chosen so that the scaling factor β need not be stored separately (Stewart, 1977). If $\gamma = \|y\|$, the value of β is:

$$\beta = \frac{1}{2}\|u\|^2 = \frac{1}{2}\,\frac{(y_1^2 + 2|y_1|\,\|y\| + \|y\|^2 + y_2^2 + \cdots + y_\ell^2)}{\|y\|^2}$$

$$= \frac{\|y\|^2 + |y_1|\,\|y\|}{\|y\|^2}$$

$$= 1 + \frac{|y_1|}{\|y\|} \quad .$$

If $\gamma = \|y\|$, then $u_1 = y_1/\|y\| + \text{sign}(y_1)$, so that $|u_1| = 1 + |y_1|/\|y\| = \beta$.

Thus, the Householder vector u is given by:

$$u_1 = \frac{y_1}{\|y\|} + \text{sign}(y_1) \qquad \text{(stored separately)}$$

$$u_j = \frac{y_j}{\|y\|}, \qquad j = 2, \ldots, \ell \qquad \text{(stored in j-th component of } y) \quad .$$

This scaling of the Householder vector eliminates the need to store β separately or recompute β. Furthermore, the Householder vector is normalized so that its length lies between $\sqrt{2}$ and 2; this latter property is useful in avoiding overflow or underflow when applying the transformation to other vectors.

The arrangement described above is termed storage of the Householder transformation <u>in compact form</u>.

```
      SUBROUTINE HMULTC ( IROW,  ICOL,  NROW,  NCOL,  NDIM,  QRV,
     1    CNORM,  UALPHA,  LERROR )
C
      INTEGER    IROW, ICOL, NROW, NCOL, NDIM, LERROR
      DOUBLE PRECISION   CNORM, UALPHA
      DOUBLE PRECISION   QRV(NDIM, NCOL)
C
C----------------------------------------------------------------------
C
C         THE SUBROUTINE HMULTC (FOR HOUSEHOLDER TRANSFORMATION
C     MULTIPLIED OVER COLUMNS) WILL CONSTRUCT THE HOUSEHOLDER
C     TRANSFORMATION DESIGNED TO ANNIHILATE ELEMENTS IROW+1 THROUGH
C     NROW OF COLUMN ICOL OF THE MATRIX QRV, AND THEN APPLY THIS
C     TRANSFORMATION TO COLUMNS ICOL+1 THROUGH NCOL OF QRV.
C
C         THE IROW-TH COMPONENT OF THE VECTOR DEFINING THE
C     HOUSEHOLDER TRANSFORMATION IS STORED IN UALPHA.  THE VECTOR
C     IS NORMALIZED SO THAT THE MAGNITUDE OF UALPHA IS ALSO THE
C     SCALING FACTOR FOR THE TRANSFORMATION (SEE THE EXTERNAL
C     DOCUMENTATION FOR FURTHER DETAILS).  COMPONENTS IROW+1
C     THROUGH NROW OF THE VECTOR ARE STORED IN THE CORRESPONDING
C     ELEMENTS OF COLUMN ICOL OF QRV.
C
C
C     THE FORMAL PARAMETERS OF HMULTC ARE:
C
C        IROW ---
C               INTEGER, INPUT ONLY.
C               IROW IS THE INDEX FOR THE STARTING ELEMENT OF THE
C               HOUSEHOLDER VECTOR.  MUST BE .GE. 1.
C
C        ICOL ---
C               INTEGER, INPUT ONLY.
C               ICOL IS THE INDEX OF THE COLUMN UPON WHICH THE HOUSEHOLDER
C               TRANSFORMATION IS BASED.  MUST BE .GE. 1.
C
C        NROW ---
C               INTEGER, INPUT ONLY.
C               NROW IS THE INDEX OF THE LAST ELEMENT IN THE COLUMNS OF QRV
C               TO BE AFFECTED BY THE HOUSEHOLDER TRANSFORMATION.
C               MUST BE .GE. IROW.
C
C        NCOL ---
C               INTEGER, INPUT ONLY.
C               NCOL IS THE INDEX OF THE LAST COLUMN TO WHICH THE HOUSE-
C               HOLDER TRANSFORMATION IS APPLIED.  MUST BE .GE. ICOL.
C
C        NDIM ---
C               INTEGER, INPUT ONLY.
C               NDIM IS THE EXTERNALLY DECLARED ROW DIMENSION OF THE
```

```
C             MATRIX QRV.  MUST BE .GE. NROW.
C
C      QRV ---
C             DOUBLE PRECISION ARRAY, OF CONCEPTUAL DIMENSION NROW
C             BY NCOL, AND DECLARED DIMENSION NDIM BY NCOL.
C             INPUT AND OUTPUT.
C             QRV IS THE MATRIX TO BE TRANSFORMED.  ON EXIT, ROWS
C             IROW THROUGH NROW OF COLUMNS ICOL THROUGH NCOL OF QRV WILL
C             HAVE BEEN ALTERED.
C
C      CNORM ---
C             DOUBLE PRECISION, INPUT ONLY.
C             CNORM IS THE EUCLIDEAN LENGTH OF THE REDUCED COLUMN ICOL
C             BEGINNING WITH ELEMENT IROW AND ENDING WITH ELEMENT NROW.
C             MUST BE .GT. 0.0D+0.
C
C      UALPHA ---
C             DOUBLE PRECISION, OUTPUT ONLY.
C             UALPHA CONTAINS THE IROW-TH ELEMENT OF THE HOUSEHOLDER
C             VECTOR.  THE ABSOLUTE VALUE OF UALPHA IS ALSO THE
C             SCALING FACTOR FOR THE TRANSFORMATION.  IF UALPHA IS EXACTLY
C             ZERO, THE TRANSFORMATION WAS SKIPPED.
C
C      LERROR ---
C             INTEGER, OUTPUT ONLY.
C             LERROR IS THE ERROR FLAG AND HAS TWO POSSIBLE VALUES.
C
C                LERROR = 0 : NORMAL RETURN, NO ERRORS FOUND.
C                LERROR = 1 : ERROR IN INPUT PARAMETERS.
C
C
C
C    *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                  SYSTEMS OPTIMIZATION LABORATORY
C                  DEPARTMENT OF OPERATIONS RESEARCH
C                  STANFORD UNIVERSITY
C                  STANFORD, CALIFORNIA 94305
C
C    *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C      DECLARATION OF LOCAL VARIABLES.
C
       INTEGER    I, ICLP1, IRWP1, KCOL, KROW
       DOUBLE PRECISION    BETA, DOTPRD, ELEM, FSIGN, GAMMA
C
C      DECLARATION OF STANDARD FUNCTIONS.
```

```
C
      DOUBLE PRECISION   DABS, DSQRT
C
C     CHECK FOR ILLEGAL VALUES OF INPUT PARAMETERS.
C
      LERROR = 1
      IF ( CNORM .LE. 0.0D+0 .OR.
     *      ICOL .LE. 0 .OR.
     *      IROW .LE. 0 .OR.
     *      NDIM .LT. NROW .OR.
     *      NCOL .LT. ICOL .OR.
     *      NROW .LT. IROW ) RETURN
      LERROR = 0
C
C-----------------------------------------------------------------------
C
C     CHECK TO SEE IF THE IROW-TH ELEMENT IS THE ONLY NON-ZERO ELEMENT
C     IN THE REDUCED COLUMN.  IF SO, RETURN WITH UALPHA = ZERO.
C
C-----------------------------------------------------------------------
C
      UALPHA = 0.0D+0
      ELEM = QRV(IROW,ICOL)
      IF (DSQRT(ELEM ** 2) .EQ. CNORM .OR. IROW .EQ. NROW) RETURN
      ESIGN = 1.0D+0
      IF (ELEM .LT. 0.0D+0)  ESIGN = -ESIGN
      UALPHA = ELEM/CNORM + ESIGN
C
C     THE IROW-TH ELEMENT IN THE TRANSFORMED COLUMN BECOMES +/- CNORM,
C     WHERE THE SIGN IS OPPOSITE TO THAT OF THE ORIGINAL IROW-TH
C     ELEMENT.
C
      QRV(IROW,ICOL) = -ESIGN*CNORM
C
C     NORMALIZE ELEMENTS IROW+1 THROUGH NROW SO THAT THE SCALING
C     FACTOR OF THE TRANSFORMATION IS THE ABSOLUTE VALUE OF THE
C     IROW-TH COMPONENT.
C
      IRWP1 = IROW + 1
      DO 400 I = IRWP1, NROW
         QRV(I,ICOL) = QRV(I,ICOL)/CNORM
  400 CONTINUE
C
C-----------------------------------------------------------------------
C
C     APPLY THE TRANSFORMATION TO THE REMAINING COLUMNS OF QRV.  IF
C     ICOL IS THE ONLY COLUMN TO BE TRANSFORMED, EXIT.
C
C-----------------------------------------------------------------------
C
```

```
      IF (ICOL .EQ. NCOL) RETURN
      ICLP1 = ICOL + 1
      BETA = DABS(UALPHA)
      DO 430 KCOL = ICLP1, NCOL
C
C        FIND THE DOT PRODUCT (DOTPRD) OF THE HOUSEHOLDER VECTOR AND
C        THE COLUMN TO BE TRANSFORMED, KCOL.  ELEMENT IROW OF THE
C        HOUSEHOLDER VECTOR IS STORED IN UALPHA AND SO IS TREATED
C        SEPARATELY.
C
         DOTPRD = UALPHA * QRV(IROW,KCOL)
         DO 410 KROW = IRWP1, NROW
               DOTPRD = DOTPRD + QRV(KROW,ICOL) * QRV(KRCW,KCOL)
  410    CONTINUE
C
C        ---------------------------------------------------------------
C
C        APPLY THE HOUSEHOLDER TRANSFORMATION TO THE COLUMN KCOL.  THIS
C        IS EQUIVALENT TO SUBTRACTING OUT A MULTIPLE OF THE HOUSEHOLDER
C        VECTOR FROM THE COLUMN.  ONLY ELEMENTS IROW THROUGH NROW OF
C        THE COLUMN ARE EXPLICITLY MODIFIED, SINCE THE FIRST (IROW-1)
C        COMPONENTS OF THE HOUSEHOLDER VECTOR ARE ZERO.  THE IROW-TH
C        COMPONENT IS TREATED SEPARATELY, SINCE THE IROW-TH COMPONENT
C        OF THE HOUSEHOLDER VECTOR IS STORED IN UALPHA.
C
C        ---------------------------------------------------------------
C
         GAMMA = DOTPRD/BETA
         QRV(IROW,KCOL) = QRV(IROW,KCOL) - GAMMA * UALPHA
         DO 420 KROW = IRWP1, NROW
            QRV(KROW,KCOL)= QRV(KROW,KCOL)- GAMMA * QRV(KROW,ICOL)
  420    CONTINUE
C
C       STATEMENT 430 ENDS THE LOOP OVER COLUMNS ICOL+1 THROUGH NCOL.
C
  430 CONTINUE
      RETURN
      END
```

6.4. Subroutine HREDL

6.4.1. Purpose

The subroutine HREDL (for Householder reduction from the left) is designed to reduce a general real matrix to upper trapezoidal form by application on the left of a special sequence of Householder transformations (the process is described in detail in Section 2.1). The principal use of HREDL will be as the first step of solving a linear least-squares problem.

6.4.2. Description of method

See Section 2.1.

6.4.3. Keywords

Householder transformation; Householder reduction to upper trapezoidal form.

6.4.4. Source language

Fortran. The code in HREDL has been checked by the PFORT verifier, and is WATFIV-compatible. The current code has been written for double precision floating-point on an IBM 360/370; the machine-dependent constant EPSMCH is set in a DATA statement, and should be altered to correspond with the given computer. All variables and functions are explicitly declared.

6.4.5. Specification and parameters

See accompanying listing.

6.4.6. Error indicators

See accompanying listing (the description of the parameter LERROR). Several precautions against underflow have already been included. However, in extreme cases it may be necessary to add further safeguards; comments in the code indicate where such changes might be made.

6.4.7. Auxiliary routines

HREDL calls the subroutines HMULTC and LENSQ. It also requires the standard functions DSQRT and IABS.

6.4.8. Program size

137 Fortran source statements.

6.4.9. Array storage

No locally declared arrays.

6.4.10. Timing

The number of arithmetic operations (additions/multiplications) necessary to reduce a full-rank m by n matrix ($m \geq n$) to trapezoidal form using Householder transformations is of approximate order $(mn^2 - n^3/3)$.

6.4.11. Further comments

Column interchanges. If a column interchange strategy is used in HREDL, the relevant columns are physically interchanged, and the exchange is recorded in the integer array IPERM, whose k-th component gives the index in the original matrix of the k-th reduced column. For example, let $n = 4$, and let the original matrix A be partitioned into its columns:

$$A = [a_1 \quad a_2 \quad a_3 \quad a_4] \ .$$

If the IPERM array is (4 2 1 3), the matrix actually reduced is:

$$AP = [a_4 \quad a_2 \quad a_1 \quad a_3] \ .$$

Definition of "size" of a remaining column. When using a column interchange strategy in transforming from the left, the "largest" remaining column is chosen to be reduced at each stage. However, it is impossible to devise a universally applicable definition of the "size" of a remaining column (see Section 4.2). Therefore, three column-based options are provided with HREDL to allow some flexibility in this definition. The size of a remaining column can be specified as:

(a) the ratio of its current length to the original length of the full column. This measure is appropriate when the elements in each column have a fairly consistent scaling with one another, but not necessarily with all other columns -- for example, if each column represents the observed values of a particular quantity, but the scaling of each quantity may vary.

(b) Its current length. This is the most obvious definition of "size," and is appropriate when a uniform scaling exists among all columns -- for example, the matrix may have been scaled before calling HREDL.

(c) The ratio of its current length to a user-provided scaling factor. This choice allows some correction to be made for badly scaled data. It should be emphasized that this option does not mean that a weighted least-squares problem is solved, but only that the user can exercise some control over which columns are considered "largest." For example, this option could be used to encourage (or discourage) the selection of particular columns in finding a linearly independent set, or in the early stages of the reduction.

Definition of "negligible". The test for whether a particular remaining column is "negligible" is of the form:

$$\text{(size of column)} \leq \text{tolerance} \quad ,$$

and thus involves the specification of a tolerance. Unfortunately, no universal magic value of such a tolerance exists; the design of HREDL accordingly includes a procedure for defining "negligible" that is based on a combination of user-provided and machine-dependent information.

First, the parameter COLEPS allows the user to control the definition of "negligible," based on known properties of the data and on the nature of the problem to be solved.

In particular, COLEPS should reflect the accuracy of the data (with respect to the chosen definition of size). For example, if the matrix consists of observed values with at most 4 figures of relative accuracy, COLEPS should be no less than 10^{-4}.

The type of problem to be solved also affects the choice of COLEPS. Roughly speaking, if the residual vector of a least-squares problem is not "small," the conditioning of the problem is proportional to the _squared reciprocal_ of the length of the smallest remaining column (see Stewart, 1973, for a precise statement and details). Thus, for numerical stability in a general least-squares problem, it is usually desirable to set COLEPS at no less than the square root of machine precision, even if the data are fully accurate. However, COLEPS may safely be set to a smaller value if it is known _a priori_ that the residual vector will be essentially zero, or if stabilizing linear constraints are to be added to the problem. Even in these cases, COLEPS should never be less than machine precision.

In addition to the information provided by COLEPS, it seems reasonable to include in the test for "negligible" some quantity based on rounding errors that occur during computation of the reduced matrix, since clearly the calculated values will differ from the exact values. The backward error analysis given in Lawson and Hanson (1974) includes the following result. Let the k Householder transformations $H_k, \ldots, H_1$ be applied to the m-vector a, and let $\tilde{a}$ be the computed transformed vector; then

$$\|\tilde{a} - H_k \cdots H_1 a\|_2 \leq (6m - 3k + 40)k\varepsilon \|a\|_2 + \mathcal{O}(\varepsilon^2) ,$$

where ε is the precision of the floating point arithmetic.

This bound is used in HREDL in the following way. At the beginning of the (k + 1)st stage of the reduction, the tolerance $\varepsilon_b(j)$ is defined for the j-th remaining column as:

$$\varepsilon_b(j) = (6m - 3k + 40)k\varepsilon \|a_j\|_2 ,$$

where $\|a_j\|_2$ is the original Euclidean length of the full column. If the length of the computed j-th remaining column is less than $\varepsilon_b(j)$, the column is considered to be "negligible." Note that if the computed length exceeds $\varepsilon_b(j)$, the remaining column could not be zero with exact arithmetic. This test is conservative, in the sense that it is based on an error bound which is inherently an overestimate; thus, a column may not be negligible even though its norm is less than the given tolerance. However, as indicated in Section 4.3, it is believed that in close cases a questionable column should be viewed as negligible, thus leading to a smaller value of the estimated rank.

The usual definition of "negligible" in HREDL involves a tolerance that is the larger of COLEPS and the tolerance based on backward error analysis. It is possible to use only the value of COLEPS, but this alternative is not recommended except in extreme cases for the expert user.

Skipping a transformation. If the current remaining column to be reduced is initially in the desired form, no Householder transformation should be applied. To test for this possibility, the squared length of the column is computed as the squared length of the subdiagonal portion, plus the squared diagonal element. If the subdiagonal portion is exactly zero, or if the computed square root of the squared length of the column is equal to the computed square root of the squared diagonal, the column is considered to be already in a suitable form, and the transformation is skipped. This is indicated by setting HVECL(k) to zero if the k-th transformation from the left is skipped.

Recalculation of squared lengths of remaining columns. When a column interchange strategy is used in HREDL, at every step the "largest" remaining column is reduced next. Any measure of the size of the column will include its length; we now consider the procedure for computing the lengths of all remaining columns.

The length of the remaining column selected to be reduced must always be re-computed, to maintain the requisite level of accuracy in the computed transformations. However, if the lengths of all remaining columns are re-computed at the k-th stage of reducing an m by n matrix $(m \geq n)$, approximately $(n - k)(m - k)$ extra additions/multiplications are required, which will increase by a factor of about 1.5 the number of operations involved in the total reduction. To avoid this additional work, it is possible in theory to obtain the squared lengths of the remaining columns at any stage by modifying their lengths at the previous stage. At the $(k + 1)$st step, the j-th remaining column is simply a transformation of j-th remaining column at the k-th step, without its first component, i.e.,

$$\begin{array}{ccc} \text{Remaining column at stage } k & \text{Transformed remaining column} & \text{Remaining column at stage } k+1 \\ z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_\ell \end{bmatrix} \rightarrow & z' = \begin{bmatrix} z'_1 \\ z'_2 \\ \vdots \\ z'_\ell \end{bmatrix} \rightarrow & \bar{z} = \begin{bmatrix} z'_2 \\ \vdots \\ z'_\ell \end{bmatrix} \end{array} .$$

Because Householder transformations preserve Euclidean length, the squared length of the transformed column will be invariant; thus, the length of the remaining column at the $(k + 1)$st stage is given by:

$$\|\bar{z}\|^2 = (z'_2)^2 + \cdots + (z'_\ell)^2 = (z'_1)^2 + \cdots + (z'_\ell)^2 - (z'_1)^2$$

$$= \|z'\|^2 - (z'_1)^2 = \|z\|^2 - (z'_1)^2 \quad ,$$

where $\|z\|^2$ is the squared length from the previous stage.

This technique could be used to update the squared column lengths throughout the entire reduction; however, in practice rounding errors may cause a serious loss of precision when the lengths are modified several times (see Stewart, 1977, for the error analysis bounds). Therefore, in HREDL the squared lengths are periodically re-computed from scratch, using the following test (which is more conservative than that given by Stewart, 1977): the lengths are re-computed whenever the ratio of the _largest_ _updated_ squared length to the largest squared length previously computed from scratch is less than $\varepsilon^{1/4}$, where ε is machine precision. Only the maximum values are compared because a possible loss of accuracy in the squared lengths of other column is unimportant; the only problem arises when the deterioration due to rounding causes uncertainty in the choice of the next column to be reduced.

```
      SUBROUTINE HREDL ( NROW,  NCOL,  NDIM,  QRV,  COLEPS,  MODTOL,
     1  INTERC,  SCALE,  NRANK,  HVECL,  IPERM,  WORK,  IERROR )
      INTEGER NROW, NCOL, NDIM, MODTOL, NRANK, IERROR
      INTEGER IPERM(NCOL)
      LOGICAL INTERC
      DOUBLE PRECISION COLEPS
      DOUBLE PRECISION QRV(NDIM, NCOL), SCALE(NCOL), HVECL(NCOL),
     1      WORK(NCOL)
C
C
C-----------------------------------------------------------------------
C
C
C      THE SUBROUTINE HREDL (FOR HOUSEHOLDER REDUCTION FROM THE LEFT)
C  IS DESIGNED TO CARRY OUT THE REDUCTION OF A GENERAL REAL NROW BY
C  NCOL MATRIX QRV TO UPPER TRAPEZOIDAL OR UPPER TRIANGULAR FORM
C  THROUGH APPLICATION OF A SEQUENCE OF ORTHOGONAL (HOUSEHOLDER)
C  TRANSFORMATIONS ON THE LEFT.  THE REDUCTION WILL TERMINATE (A) WHEN
C  MIN(NROW,NCOL) COLUMNS HAVE BEEN SUCCESSFULLY REDUCED, OR (B) WHEN
C  THE NEXT COLUMN TO BE REDUCED IS 'NEGLIGIBLE', USING THE MEASURE OF
C  'SIZE' INDICATED BY THE INPUT FLAG 'MODTOL' (SEE BELOW).  IF COLUMN
C  INTERCHANGES ARE USED DURING THE REDUCTION, THE NUMBER OF COLUMNS
C  THAT WERE SUCCESSFULY REDUCED PROVIDES AN ESTIMATE OF THE RANK OF
C  THE ORIGINAL MATRIX.
C
C      HREDL MAY BE USED TO FIND A LEAST-SQUARES SOLUTION OF A GENERAL
C  LINEAR SYSTEM (SQUARE, OVER-DETERMINED, OR UNDER-DETERMINED).  IF
C  THE ORIGINAL MATRIX IS FOUND TO BE RANK-DEFICIENT, THE ACCOMPANYING
C  SUBROUTINE HREDR CAN BE USED TO CARRY OUT A FURTHER REDUCTION BY
C  APPLYING HOUSEHOLDER TRANSFORMATIONS ON THE RIGHT; IN THIS CASE, THE
C  MINIMUM-LENGTH LEAST SQUARES SOLUTION CAN BE COMPUTED.  THE OUTPUT
C  OF HREDL MAY BE USED AS INPUT TO THE SUBROUTINE FRMORT, TO OBTAIN AN
C  EXPLICIT REPRESENTATION OF A PORTION OF THE COMPLETE ORTHOGONAL
C  FACTORIZATION OF THE ORIGINAL MATRIX.
C
C      THE OPTIONS AVAILABLE WITH HREDL ARE OF TWO KINDS:
C
C  (1) OPTIONAL COLUMN INTERCHANGES.
C      IT IS OFTEN DESIRABLE TO INTERCHANGE COLUMNS AS THE
C  REDUCTION FROM THE LEFT IS CARRIED OUT (SEE THE ACCOMPANYING
C  DOCUMENTATION FOR A FULL DISCUSSION).  THE INTERCHANGE STRATEGY
C  SELECTS THE 'LARGEST' UNREDUCED COLUMN AS THE NEXT COLUMN TO BE
C  REDUCED, BASED ON AN APPROPRIATE MEASURE OF 'SIZE', AS DISCUSSED IN
C  (2) BELOW.  THE USER MAY SPECIFY WHETHER OR NOT INTERCHANGES ARE TO
C  BE MADE BY SETTING THE LOGICAL FLAG 'INTERC' TO 'TRUE' OR 'FALSE'.
C
C  (2) DEFINITION OF 'SIZE', SCALING
C      THE 'SIZE' OF A UNREDUCED COLUMN MUST BE MEASURED IN ORDER TO
C  DECIDE WHICH COLUMN IS THE 'LARGEST', AND WHEN A COLUMN IS
C  'NEGLIGIBLE'.  HREDL ALLOWS THE FOLLOWING THREE MEASURES OF THE SIZE
```

```
C   OF A COLUMN:
C      (A) THE RATIO OF ITS CURRENT LENGTH TO ITS ORIGINAL LENGTH
C      (A RELATIVE MEASURE);
C      (B) ITS CURRENT LENGTH (AN ABSOLUTE MEASURE);
C      (C) THE RATIO OF ITS CURRENT LENGTH TO A POSITIVE SCALING
C      FACTOR PROVIDED BY THE USER.
C
C
C      INFORMATION CONCERNING THE HOUSEHOLDER TRANSFORMATIONS APPLIED
C   FROM THE LEFT IS STORED BELOW THE DIAGONAL OF QRV, AND IN THE
C   AUXILIARY VECTOR HVECT (FULL DETAILS ARE GIVEN IN THE EXTERNAL
C   DOCUMENTATION).  THE UPPER TRAPEZOID TO WHICH THE ORIGINAL MATRIX
C   HAS BEEN REDUCED IS STORED IN THE CORRESPONDING PORTION OF QRV.
C   COLUMN INTERCHANGES ARE RECORDED IN THE INTEGER ARRAY IPERM.
C
C
C
C   THE FORMAL PARAMETERS OF QRVFCT ARE:
C
C   NROW -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF ROWS OF THE MATRIX QRV.  MUST BE .GT. 0.
C
C   NCOL -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF COLUMNS OF THE MATRIX QRV.  MUST BE .GT. 0.
C
C   NDIM -
C         INTEGER, INPUT ONLY.
C         THE DECLARED ROW DIMENSION OF THE MATRIX QRV IN THE CALLING
C         SUBPROGRAM.  MUST BE .GE. NROW.
C
C   QRV -
C         DOUBLE PRECISION ARRAY, OF CONCEPTUAL DIMENSION NROW BY NCOL,
C         AND DECLARED DIMENSION NDIM BY NCOL; INPUT AND OUTPUT.
C         ON INPUT, QRV IS THE MATRIX TO BE REDUCED TO UPPER TRAPEZOIDAL
C         FORM.  ON EXIT, THE DESIRED UPPER TRAPEZOID IS STORED IN THE
C         CORRESPONDING PORTION OF QRV.  COMPONENTS K+1 THROUGH NROW
C         OF THE VECTOR ASSOCIATED WITH THE K-TH HOUSEHOLDER TRANS-
C         FORMATION ARE STORED BELOW THE DIAGONAL IN THE K-TH COLUMN
C         OF QRV.  NOTE THAT THE COLUMNS OF THE TRAPEZOID MAY CORRESPOND
C         TO A PERMUTATION OF THE COLUMNS OF THE ORIGINAL MATRIX.
C
C
C   COLEPS -
C         DOUBLE PRECISION, INPUT ONLY.
C         COLEPS IS A VALUE TO BE USED IN DETERMINING WHETHER THE SIZE OF
C         THE NEXT COLUMN TO BE REDUCED IS 'NEGLIGIBLE' (SEE THE
C         DISCUSSION UNDER 'MODCOL' FOR A DEFINITION OF 'SIZE').  GREAT
C         CARE SHOULD BE EXERCISED IN CHOOSING COLEPS, AND THE USER IS
```

```
C       URGED TO READ THE EXTERNAL DOCUMENTATION ON THIS SUBJECT VERY
C       CAREFULLY.  COLEPS SHOULD BE NO SMALLER THAN THE UNCERTAINTY IN
C       THE DATA, RELATIVE TO THE APPROPRIATE SCALING FACTOR AS
C       MENTIONED UNDER 'MODTOL'.  COLEPS SHOULD BE NO SMALLER THAN
C       MACHINE PRECISION EXCEPT IN EXTREME CIRCUMSTANCES.  IF HREDL IS
C       USED IN CONJUNCTION WITH AN UNCONSTRAINED LEAST-SQUARES
C       PROBLEM, THEN IT IS FREQUENTLY WORTHWHILE TO USE A CONSERVATIVE
C       VALUE FOR COLEPS (E.G., THE SQUARE ROOT OF MACHINE PRECISION)
C       TO AVOID NUMERICAL DIFFICULTIES.
C
C  MODTOL -
C       INTEGER, INPUT ONLY.
C       MODTOL IS AN INTEGER FLAG WITH TWO PURPOSES: (1) THE ABSOLUTE
C       VALUE OF MODTOL INDICATES THE MEASURE OF 'SIZE' TO BE USED FOR
C       EACH COLUMN; (2) THE SIGN OF MODTOL INDICATES THE DEFINITION OF
C       'NEGLIGIBLE'.
C
C       IF MODTOL = 1 OR -1, THE SIZE OF A REMAINING COLUMN IS
C       DEFINED AS THE RATIO OF ITS CURRENT LENGTH TO THE ORIGINAL
C       LENGTH OF THE FULL COLUMN.
C
C       IF MODTOL = 2 OR -2, THE SIZE OF A REMAINING COLUMN IS
C       DEFINED AS ITS CURRENT LENGTH.
C
C       IF MODTOL = 3 OR -3, THE SIZE OF A REMAINING COLUMN IS
C       DEFINED AS THE RATIO OF ITS CURRENT LENGTH TO THE SCALING
C       FACTOR PROVIDED BY THE USER FOR THAT COLUMN.
C
C       POSITIVE VALUES OF MODTOL INDICATE THAT THE TEST FOR
C       'NEGLIGIBLE' IS BASED ON THE GREATER OF THE USER-PROVIDED
C       TOLERANCE COLEPS, AND A QUANTITY BASED ON THE BACKWARD ERROR
C       ANALYSIS OF HOUSEHOLDER TRANSFORMATIONS TO A VECTOR (SEE
C       LAWSON AND HANSON, SOLVING LEAST-SQUARES PROBLEMS, PRENTICE-
C       HALL, 1974, CHAPTER 15, FOR DETAILS).   NEGATIVE VALUES OF
C       MODTOL INDICATE THAT ONLY COLEPS IS TO BE USED AS A TEST FOR
C       'NEGLIGIBLE'.  IF MODTOL IS NEGATIVE, COLEPS MUST BE NON-
C       NEGATIVE.
C
C
C  INTERC -
C       LOGICAL, INPUT ONLY.
C       IF INTERC IS TRUE, THE COLUMN INTERCHANGE STRATEGY DESCRIBED
C       ABOVE WILL BE USED.  IF INTERC IS FALSE, THE COLUMNS WILL BE
C       REDUCED IN THE GIVEN ORDER.
C
C
C  SCALE -
C       DOUBLE PRECISION VECTOR, OF LENGTH NCOL.  OPTIONAL INPUT, AND
C       OUTPUT.
C       DURING THE REDUCTION, THE SIZE OF THE J-TH COLUMN IS DEFINED AS
```

```
C        THE RATIO OF ITS CURRENT LENGTH TO SCALE(J).  IF MODTOL = 1, -1,
C        2, OR -2, THE USER NEED NOT INITIALIZE THE SCALE ARRAY.  IF
C        MODTOL = 1 OR -1, THE SCALE ARRAY WILL CONTAIN ON EXIT THE
C        ORIGINAL SQUARED LENGTHS OF THE NON-ZERO ORIGINAL COLUMNS.
C        IF MODTOL = 2 OR -2, EACH ELEMENT OF THE SCALE ARRAY WILL BE
C        CONTAIN UNITY ON EXIT.  IF MODTOL = 3 OR -3, EACH COMPONENT
C        SCALE(J) MUST BE SET ON ENTRY TO A POSITIVE VALUE TO BE USED AS
C        A SCALING FACTOR IN MEASURING THE SIZE OF THE J-TH COLUMN.  THE
C        SCALE ARRAY WILL BE RE-ORDERED DURING EXECUTION OF HREDL, TO
C        CORRESPOND WITH ANY COLUMN INTERCHANGES.
C
C   NRANK -
C        INTEGER, OUTPUT ONLY.
C        NRANK IS THE NUMBER OF COLUMNS THAT WERE SUCCESSFULLY REDUCED.
C        IF INTERCHANGES WERE USED, NRANK IS AN ESTIMATE OF THE NUMERICAL
C        RANK OF THE ORIGINAL MATRIX.
C
C   HVECL -
C        DOUBLE PRECISION VECTOR, OF LENGTH NCOL.  OUTPUT ONLY.
C        ON EXIT, HVECL(K), K = 1, NRANK, WILL CONTAIN THE K-TH
C        COMPONENT OF THE VECTOR CORRESPONDING TO THE K-TH HOUSEHOLDER
C        TRANSFORMATION.  EACH VECTOR IS NORMALIZED SO THAT THE ABSOLUTE
C        VALUE OF HVECL(K) IS ALSO THE SCALING FACTOR FOR THE K-TH
C        TRANSFORMATION.  IF HVECL(K) IS ZERO, THEN THE K-TH
C        TRANSFORMATION WAS SKIPPED.  NOTE THAT THE INDEXING OF HVECL
C        CORRESPONDS TO THE ORDER IN WHICH THE COLUMNS WERE ACTUALLY
C        REDUCED, I.E., HVECL(1) CONTAINS THE FIRST COMPONENT OF THE
C        FIRST HOUSEHOLDER VECTOR.  THE HVECL ARRAY IS ALSO USED WITHIN
C        HREDL TO STORE THE SQUARED LENGTHS OF THE UNREDUCED COLUMNS.
C
C   IPERM -
C        INTEGER VECTOR, OF LENGTH NCOL.  OUTPUT ONLY.
C        ANY COLUMN INTERCHANGES INDICATED DURING THE REDUCTION FROM THE
C        LEFT ARE EXPLICITLY CARRIED OUT, AND ARE RECORDED IN THE ARRAY
C        IPERM.  IPERM(K) GIVES THE INDEX IN THE ORIGINAL MATRIX OF THE
C        K-TH COLUMN THAT WAS REDUCED; IF INTERC IS FALSE, IPERM(K) = K.
C        IT IS IMPORTANT TO NOTE THAT THE INDEXING OF THE VECTORS HVECL
C        AND SCALE ON EXIT IS BASED ON THE ORDER IN WHICH THE COLUMNS
C        WERE REDUCED, AND NOT ON THE ORIGINAL COLUMN ORDERING.
C
C   WORK -
C        DOUBLE PRECISION VECTOR, OF LENGTH NCOL.  OUTPUT ONLY.
C        THE WORK ARRAY IS USED FOR INTERMEDIATE STORAGE DURING
C        EXECUTION OF QRVECT.  UPON TERMINATION, WORK(K) GIVES THE
C        ORIGINAL LENGTH OF THE K-TH COLUMN THAT WAS REDUCED.
C
C   LERROR -
C        INTEGER, OUTPUT ONLY.
C        AN ERROR FLAG, WHOSE POSSIBLE VALUES ARE AS FOLLOWS.
C
```

```
C       IERROR = 0 : NORMAL TERMINATION, NO ERRORS.
C
C       IERROR = 1 : EITHER NROW OR NCOL IS NON-POSITIVE, OR MODTOL
C       IS NEGATIVE AND COLEPS IS NEGATIVE.
C
C       IERROR = 2 : MODTOL = 3 OR -3, AND ONE OF THE ELEMENTS OF THE
C       SCALE ARRAY IS NON-POSITIVE.
C
C       IERROR = 3 : INTERC IS TRUE, AND ALL COLUMNS OF THE INPUT
C       MATRIX ARE IDENTICALLY ZERO.
C
C       IERROR = 4 : INTERC IS FALSE, AND AT SOME STAGE THE NEXT COLUMN
C       TO BE REDUCED WAS 'NEGLIGIBLE'.
C
C       IERROR > 10 : AN ERROR OCCURRED DURING EXECUTION OF HMULTC.
C       THIS SHOULD NEVER HAPPEN.  IF IT DOES, CHECK THE ERROR IN HMULTC
C       CORRESPONDING TO THE VALUE (IERROR-10).
C
C
C   *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                 SYSTEMS OPTIMIZATION LABORATORY
C                 DEPARTMENT OF OPERATIONS RESEARCH
C                 STANFORD UNIVERSITY
C                 STANFORD, CALIFORNIA 94305
C
C   *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C
C      DECLARATION OF LOCAL VARIABLES.
C
       INTEGER   I, IERR, ISAVE, J, K, KP1, LEN, MAXCOL, MNMIN, MODE
       DOUBLE PRECISION   CNORM, COLTOL, EPSMCH, ERRCT, QEPS, RMAX,
      1    RTEPS, RTEST, SAVEMX, SSAVE, TOLRCK, VLENSC, VSDIAG
C
C      DECLARATION OF STANDARD FUNCTIONS.
C
       INTEGER   IABS
       DOUBLE PRECISION   DSQRT
C
C   EPSMCH IS THE RELATIVE PRECISION OF THE FLOATING POINT ARITHMETIC.
C
       DATA EPSMCH / 2.22D-16 /
C
       RTEPS = DSQRT(EPSMCH)
       QEPS = DSQRT(RTEPS)
C
```

```
      IERROR = 1
C
C-----------------------------------------------------------------------
C
C   CHECK FOR INVALID VALUES OF INPUT PARAMETERS.
C
C-----------------------------------------------------------------------
C
      IF (NROW .LE. 0 .OR. NCOL .LE. 0)   RETURN
      IF (MODTOL .LT. 0 .AND. COLEPS .LT. 0.0D+0) RETURN
C
C   FIND THE MINIMUM OF (NROW, NCOL) AND INITIALIZE THE IPERM ARRAY.
C
      IF (NROW .GE. NCOL)  GO TO 10
      MNMIN = NROW
      GO TO 20
   10 MNMIN = NCOL
   20 CONTINUE
      DO 30 I = 1, NCOL
         IPERM(I) = I
   30 CONTINUE
      NRANK = 0
C
C-----------------------------------------------------------------------
C
C   THE ORIGINAL SQUARED COLUMN LENGTHS ARE COMPUTED AND STORED IN THE
C   HVECL ARRAY.  THE ORIGINAL LENGTHS ARE STORED IN THE WORK ARRAY, AND
C   RETAINED THROUGHOUT EXECUTION OF HREDL, TO BE USED IN CONJUNCTION
C   WITH TESTS INVOLVING BACKWARD ERROR ANALYSIS BOUNDS.
C
C-----------------------------------------------------------------------
C
      DO 40 J = 1, NCOL
         CALL LENSQ(NROW, QRV(1,J), HVECL(J))
         WORK(J) = DSQRT(HVECL(J))
   40 CONTINUE
      MODE = IABS(MODTOL)
      GO TO (50, 70, 90),  MODE
C
C
C-----------------------------------------------------------------------
C
C   WHEN MODTOL = 1 OR -1, THE ORIGINAL COLUMN LENGTHS SERVE AS SCALING
C   FACTORS.
C
C-----------------------------------------------------------------------
C
   50 CONTINUE
      DO 60 J = 1, NCOL
         SCALE(J) = WORK(J)
         IF (WORK(J) .EQ. 0.0D+0)   SCALE(J) = 1.0D+0
   60 CONTINUE
      GO TO 110
C
C-----------------------------------------------------------------------
C
C  WHEN MODTOL = 2 OR -2, AN ABSOLUTE CRITERION OF SIZE IS USED, AND
C  THE SCALING FACTORS ARE SET TO UNITY.
C
C-----------------------------------------------------------------------
C
   70 CONTINUE
      DO 80 J = 1, NCOL
         SCALE(J) = 1.0D+0
   80 CONTINUE
      GO TO 110
C
C-----------------------------------------------------------------------
C
C  WHEN MODTOL = 3 OR -3, THE USER-SUPPLIED SCALE FACTORS ARE USED.
C  IF ANY SCALE(J) IS NON-POSITIVE, AN ERROR EXIT IS TAKEN.
C
C-----------------------------------------------------------------------
C
   90 CONTINUE
      IERROR = 2
      DO 100 J = 1, NCOL
         IF (SCALE(J) .LE. 0.0D+0)   RETURN
  100 CONTINUE
C
  110 CONTINUE
C
      MAXCOL = 1
C
C     IF NO INTERCHANGES ARE TO BE CARRIED OUT, THE FIRST COLUMN IS
C     REDUCED FIRST.
C
      IF (.NOT. INTERC)   GO TO 130
C
C-----------------------------------------------------------------------
C
C  IF INTERCHANGES ARE TO BE CARRIED OUT, DETERMINE THE COLUMN OF
C  MAXIMUM SCALED LENGTH.
C
C-----------------------------------------------------------------------
C
      RMAX = 0.0D+0
      DO 120 J = 1, NCOL
         RTEST = WORK(J)/SCALE(J)
         IF (RTEST .LE. RMAX)   GO TO 120
```

```
            RMAX = RTEST
            MAXCOL = J
  120 CONTINUE
      LERROR = 3
      IF (RMAX .EQ. 0.0D+0)  RETURN
      SAVEMX = RMAX
  130 CONTINUE
C
C
C-------------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 300 IS THE MAIN LOOP OVER THE COLUMNS,
C  TO REDUCE THE MATRIX TO UPPER TRIANGULAR FORM BY APPLICATION OF
C  HOUSEHOLDER TRANSFORMATIONS FROM THE LEFT.
C
C-------------------------------------------------------------------------
C
C
      DO 300 K = 1, MNMIN
C
C        TEST WHETHER A COLUMN INTERCHANGE IS TO BE CARRIED OUT.
C
         IF (MAXCOL .EQ. K)  GO TO 210
C
C        INTERCHANGE COLUMNS K AND MAXCOL OF QRV, AND THE CORRESPONDING
C        ELEMENTS OF THE HVECL, SCALE AND WORK ARRAYS.  RECORD THE
C        INTERCHANGE IN THE IPERM ARRAY, WHOSE K-TH ELEMENT GIVES THE
C        INDEX IN THE ORIGINAL MATRIX OF THE K-TH COLUMN TO BE REDUCED.
C
         ISAVE = IPERM(K)
         IPERM(K) = IPERM(MAXCOL)
         IPERM(MAXCOL) = ISAVE
         SSAVE = HVECL(K)
         HVECL(K) = HVECL(MAXCOL)
         HVECL(MAXCOL) = SSAVE
         SSAVE = SCALE(K)
         SCALE(K) = SCALE(MAXCOL)
         SCALE(MAXCOL) = SSAVE
         SSAVE = WORK(K)
         WORK(K) = WORK(MAXCOL)
         WORK(MAXCOL) = SSAVE
         DO 200 I = 1, NROW
            SSAVE = QRV(I,K)
            QRV(I,K) = QRV(I,MAXCOL)
            QRV(I,MAXCOL) = SSAVE
  200    CONTINUE
C
C
  210    CONTINUE
C
```

```
C        COMPUTE THE SQUARED LENGTH OF THE COLUMN TO BE REDUCED IN TWO
C        PARTS, WITH A SEPARATE CALCULATION OF THE SUBDIAGONAL PORTION.
C
         VSDIAG = 0.0D+0
         IF (K .EQ. NROW)  GO TO 220
         LEN = NROW - K
         CALL LENSQ(LEN, QRV(K+1,K), VSDIAG)
C
C        FURTHER PROTECTION AGAINST UNDERFLOW CAN BE INCLUDED HERE IF
C        NECESSARY.
C
  220    VLENSQ = VSDIAG + QRV(K,K)**2
C
C        TEST WHETHER THE COLUMN TO BE REDUCED IS 'NEGLIGIBLE', USING
C        THE CRITERION DETERMINED BY THE VALUE OF MODTOL.
C
C
C        THE ERROR BOUND BASED ON THE BACKWARD ERROR ANALYSIS REQUIRES
C        THE ORIGINAL COLUMN LENGTHS, WHICH ARE STORED IN THE WORK
C        ARRAY.
C
         IERR = (K-1)*(6*NROW - 3*(K-1) + 40)
         ERFCT = IERR
         TOLBCK = ERFCT*EPSMCH*WORK(K)
         COLTOL = COLEPS*SCALE(K)
C
C        IF MODTOL .GT. 0, USE THE LARGER OF COLTOL AND THE BACKWARD
C        ERROR BOUND AS A TOLERANCE.  OTHERWISE, USE ONLY COLTOL.
C
         IF (MODTOL .GT. 0 .AND. TOLBCK .GT. COLTOL)  COLTOL = TOLBCK
         CNORM = DSQRT(VLENSQ)
         IF (CNORM .LE. COLTOL)  GO TO 310
C
C        ------------------------------------------------------------
C
C        THE K-TH COLUMN IS NOT 'NEGLIGIBLE'.  INCREMENT THE RANK, AND
C        PROCEED WITH THE REDUCTION.
C
C        ------------------------------------------------------------
C
         NRANK = NRANK + 1
         HVECL(K) = 0.0D+0
C
C        TEST WHETHER THE K-TH TRANSFORMATION CAN BE SKIPPED.
C
         IF (K .EQ. NROW)  GO TO 300
         IF (VSDIAG .EQ. 0.0D+0)  GO TO 230
C
C        ------------------------------------------------------------
C
```

```
C          THE SUBROUTINE HMULTC CONSTRUCTS THE HOUSEHOLDER TRANSFORMATION
C          THAT ANNIHILATES THE SUB-DIAGONAL ELEMENTS OF COLUMN K AND
C          LEAVES ELEMENTS 1 THROUGH K-1 UNALTERED.  IT THEN APPLIES
C          THIS TRANSFORMATION TO THE REMAINING COLUMNS OF QRV.
C          THE TRANSFORMATION MAY BE WRITTEN AS I - U*U**T/BETA, WHERE
C          THE FIRST (K-1) COMPONENTS OF THE VECTOR U ARE ZERO.  HVECL(K)
C          CONTAINS THE K-TH COMPONENT OF U FOR THE K-TH TRANSFORMATION,
C          WHICH HAS BEEN NORMALIZED SO THAT ABS(HVECL(K)) CONTAINS THE
C          VALUE OF BETA.
C
C          ------------------------------------------------------------
C
           CALL HMULTC( K, K, NROW, NCOL, NDIM, QRV, CNORM, HVECL(K),
     1           IERROR )
           IF (IERROR .NE. 0)  GO TO 320
C
  230      CONTINUE
           IF (K .EQ. NCOL)  GO TO 300
           IF (INTERC)  GO TO 240
C
C          IF NO INTERCHANGES ARE TO BE CARRIED OUT, SET MAXCOL TO K+1 IN
C          PREPARATION FOR THE NEXT TIME THROUGH THE LOOP.
C
           MAXCOL = K + 1
           GO TO 300
C
  240      CONTINUE
C
C         UPDATE THE SQUARED LENGTHS OF THE REMAINING COLUMNS, AND OBTAIN
C         THE TENTATIVE INDEX OF THE REDUCED COLUMN OF MAXIMUM SCALED
C         LENGTH.
C
           KP1 = K + 1
           RMAX = 0.0D+0
           DO 250 J = KP1, NCOL
              HVECL(J) = HVECL(J) - QRV(K,J)**2
              IF (HVECL(J) .LE. 0.0D+0)  GO TO 250
              RTEST = DSQRT(HVECL(J))/SCALE(J)
              IF (RTEST .LE. RMAX)  GO TO 250
              RMAX = RTEST
              MAXCOL = J
  250      CONTINUE
C
C          TEST WHETHER THE LENGTHS SHOULD BE RECOMPUTED, BY COMPARING
C          THE SCALED MAXIMUM WITH THE ANALOGOUS VALUE THE LAST TIME THE
C          LENGTHS WERE COMPUTED FROM SCRATCH.
C
           IF (RMAX .GT. CEPS*SAVEMX)  GO TO 300
           LEN = NROW - K
           RMAX = 0.0D+0
```

```
          DO 260 J = KP1, NCOL
             CALL LENSQ(LEN, QRV(KP1,J), HVECL(J))
             RTEST = DSQRT(HVECL(J))/SCALE(J)
             IF (RTEST .LE. RMAX)  GO TO 260
             RMAX = RTEST
             MAXCOL = J
  260     CONTINUE
          SAVEMX = RMAX
C
C-----------------------------------------------------------------------
C
C  STATEMENT 300 ENDS THE MAIN LOOP OVER THE COLUMNS
C
C-----------------------------------------------------------------------
C
  300 CONTINUE
C
C-----------------------------------------------------------------------
C
C  TO EXIT HERE, ALL MNMIN COLUMNS HAVE BEEN CONSIDERED TO BE 'NON-ZERO'
C
C-----------------------------------------------------------------------
C
      IERROR = 0
      RETURN
C
C-----------------------------------------------------------------------
C
C  AT STATEMENT 310, AT SOME STAGE OF THE REDUCTION FROM THE LEFT, THE
C  NEXT COLUMN TO BE REDUCED WAS CONSIDERED 'NEGLIGIBLE'.
C
C-----------------------------------------------------------------------
C
  310 CONTINUE
      IERROR = 4
      IF (.NOT. INTERC)  RETURN
      IERROR = 0
      RETURN
C
C  AT STATEMENT 320, IERROR .NE. 0 ON RETURN FROM HMULTC.
C
  320 IERROR = IERROR + 10
      RETURN
      END
```

6.5. Subroutine HREDR

6.5.1. Purpose

The subroutine HREDR (for Householder reduction from the right) reduces an r by n $(n > r)$ upper trapezoidal matrix to an r by r upper triangle followed by a block of zeros by applying Householder transformations on the right. This technique is used to compute the minimum-length least-squares solution and to form the complete orthogonal factorization.

6.5.2. Description of method

See Section 3.2.

6.5.3. Keywords

Householder transformation; orthogonal transformation; complete orthogonal factorization; minimum-length least-squares solution.

6.5.4. Source language

Fortran. The code in HREDR has been checked by the PFORT verifiers, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.5.5. Specification and parameters

See accompanying listing.

6.5.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.5.7. Auxiliary routines

HREDR calls the standard functions DABS and DSQRT.

6.5.8. Program size

51 Fortran source statements.

6.5.9. Array storage

There are no locally declared arrays.

6.5.10. Timing

The number of operations to carry out the reduction includes approximately $(n - r)r^2$ additions/multiplications, r square roots, and $r(n - r)$ divisions.

6.5.11. Further comments

The Householder transformations applied on the right are represented by vectors stored in a manner analogous to that described in Section 6.4.11: components $r + 1$ through n of the transformation that reduces the j-th row to appropriate form are stored in those same components of row j, and the j-th component is stored separately (in HVECR(j)).

As in HREDL, the Householder vectors are normalized, in this case so that the magnitude of the j-th component is the scaling factor for the transformation. The j-th transformation is skipped if the j-th row is already in a suitable form, based on tests as in the reduction from the left (in this case, HVECR(j) is set to zero).

```
      SUBROUTINE HREDR ( NRANK, NCOL, NDIM, QRV, VALPHA, LERROR )
      INTEGER   NRANK, NCOL, NDIM, LERROR
      DOUBLE PRECISION   QRV(NDIM, NCOL) ,  VALPHA(NCOL)
C
C
C-----------------------------------------------------------------------
C
C        THE SUBROUTINE HREDR (FOR HOUSEHOLDER REDUCTION FROM THE
C     RIGHT) IS DESIGNED TO REDUCE AN UPPER TRAPEZOIDAL MATRIX TO UPPER
C     TRIANGULAR FORM BY APPLICATION OF APPROPRIATELY CHOSEN HOUSEHOLDER
C     TRANSFORMATIONS ON THE RIGHT.
C
C        THE INPUT MATRIX, QRV, IS ASSUMED TO CONTAIN AN NRANK BY NCOL
C     UPPER TRAPEZOIDAL MATRIX IN ITS FIRST NRANK ROWS.  IN A BACKWARD
C     SWEEP THROUGH THE ROWS, I = NRANK STEP -1 UNTIL 1, THE I-TH
C     HOUSEHOLDER TRANSFORMATION IS CONSTRUCTED TO ANNIHILATE ELEMENTS
C     NRANK+1 THROUGH NCOL OF ROW I, ALTER THE (I,I) DIAGONAL ELEMENT,
C     AND LEAVE ELEMENTS 1 THROUGH I-1 AND I+1 THROUGH NRANK UNALTERED.
C     EACH SUCH TRANSFORMATION IS THEN APPLIED TO ROWS 1 THROUGH I-1.
C     THIS PROCEDURE IS DESCRIBED IN GREATER DETAIL IN LAWSON AND HANSON,
C     SOLVING LEAST-SQUARES PROBLEMS, PRENTICE-HALL, 1974.
C
C        HREDR WILL NORMALLY BE USED IN CONJUNCTION WITH HREDL, TO
C     OBTAIN THE COMPLETE ORTHOGONAL FACTORIZATION OF A GENERAL REAL
C     MATRIX, AND TO COMPUTE THE MINIMUM-LENGTH LEAST-SQUARES SOLUTION.
C
C
C     THE FORMAL PARAMETERS OF HREDR ARE:
C
C         NRANK ---
C                 INTEGER, INPUT ONLY.
C                 NRANK IS THE PREVIOUSLY DETERMINED RANK OF THE INPUT MATRIX
C                 QRV.  NRANK GIVES THE NUMBER OF TRANSFORMATIONS APPLIED
C                 ON THE RIGHT, AND THE SIZE OF THE FINAL UPPER TRIANGULAR
C                 MATRIX.
C                 MUST BE .GE. 1.
C
C         NCOL ---
C                 INTEGER, INPUT ONLY.
C                 NCOL IS THE COLUMN DIMENSION OF THE MATRIX QRV.
C                 MUST BE .GE. NRANK.
C
C         NDIM ---
C                 INTEGER, INPUT ONLY.
C                 NDIM IS THE EXTERNALLY DECLARED ROW DIMENSION OF THE MATRIX
C                 QRV.  MUST BE .GE. NRANK.
C
C         QRV ---
C                 DOUBLE PRECISION MATRIX, OF DECLARED DIMENSION NDIM BY NCOL.
C                 INPUT AND OUTPUT.
```

```
C           ON INPUT, QRV IS ASSUMED TO CONTAIN AN UPPER TRAPEZOIDAL
C           MATRIX IN ITS FIRST NRANK ROWS, AND ALL SUB-DIAGONAL
C           ELEMENTS ARE IGNORED.  ON OUTPUT, THE DESIRED UPPER
C           TRIANGULAR MATRIX IS STORED IN THE UPPER TRIANGLE OF QRV,
C           AND COMPONENTS NRANK+1 THROUGH NCOL OF THE I-TH HOUSEHOLDER
C           VECTOR ARE STORED IN COLUMNS NRANK+1 THROUGH NCOL OF ROW I.
C
C     VALPHA ---
C           DOUBLE PRECISION ARRAY, OF DIMENSION NRANK.
C           OUTPUT ONLY.
C           VALPHA(I) CONTAINS THE I-TH COMPONENT OF THE VECTOR THAT
C           DEFINES THE HOUSEHOLDER TRANSFORMATION THAT REDUCES ROW
C           I TO THE CORRECT FORM.
C
C     LERROR ---
C           INTEGER, OUTPUT ONLY.
C           LERROR IS THE ERROR FLAG WITH 4 POSSIBLE VALUES.
C
C              LERROR = 0 : NORMAL RETURN, NO ERRORS FOUND.
C              LERROR = 1 : ILLEGAL INPUT PARAMETER.
C              LERROR = 2 : THE DIAGONAL AND ALL ELEMENTS TO THE RIGHT
C                           ARE ZERO IN ONE OF THE FIRST NRANK ROWS.
C
C
C
C    *** AUTHORS:   MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                   SYSTEMS OPTIMIZATION LABORATORY
C                   DEPARTMENT OF OPERATIONS RESEARCH
C                   STANFORD UNIVERSITY
C                   STANFORD, CALIFORNIA 94305
C
C    *** DATE:      DECEMBER 1977
C
C
C------------------------------------------------------------------------
C
C
C     DECLARATION OF LOCAL VARIABLES.
C
      INTEGER  I, II, IM1, J, K, NRNKP1
      DOUBLE PRECISION  DOTPRD, ELEM, ESIGN, GAMMA, RNORM, SUM
C
C     DECLARATION OF STANDARD FUNCTIONS.
C
      DOUBLE PRECISION  DABS, DSQRT
C
C     TEST FOR ILLEGAL VALUES OF INPUT PARAMETERS.
C
      LERROR = 1
      IF (NRANK .LE. 0)  RETURN
      DO 500 I = 1, NRANK
```

```
            VALPHA(I) = 0.0D+0
  500   CONTINUE
        IF (NRANK .GT. NDIM .OR. NRANK .GT. NCOL)  RETURN
        IERROR = 0
        IF (NRANK .EQ. NCOL) RETURN
C
C----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 560 CARRIES OUT A BACKWARD SWEEP OVER THE
C  NRANK ROWS, I = NRANK STEP -1 UNTIL 1.
C
C----------------------------------------------------------------------
C
        NRNKP1 = NRANK + 1
        IERROR = 2
        DO 560 II = 1, NRANK
            I = NRANK - II + 1
C
C           COMPUTE THE LENGTH OF THE VECTOR CONSISTING OF THE DIAGONAL
C           ELEMENT AND ELEMENTS NRANK+1 THROUGH NCOL OF ROW I.  IT IS
C           COMPUTED IN TWO PARTS TO CHECK WHETHER THE TRANSFORMATION CAN
C           BE SKIPPED.
C
            SUM = 0.0D+0
            DO 510 J = NRNKP1, NCOL
               SUM = SUM + QRV(I,J) ** 2
  510       CONTINUE
            IF (SUM .EQ. 0.0D+0)  GO TO 560
            SUM = SUM + QRV(I,I)**2
            RNORM = DSQRT(SUM)
C
C           TEST WHETHER RNORM IS ZERO.  IF SO, RETURN WITH IERROR = 2.
C
            IF (RNORM .EQ. 0.0D+0)  RETURN
C
C           THE FIRST NON-ZERO ELEMENT OF THE HOUSEHOLDER VECTOR IS
C           ELEMENT I, WHICH IS STORED IN VALPHA(I).
C
            ELEM = QRV(I,I)
            ESIGN = 1.0D+0
            IF (ELEM .LT. 0.0D+0)  ESIGN = -ESIGN
            VALPHA(I) = ELEM/RNORM + ESIGN
C
C           THE TRANSFORMED DIAGONAL ELEMENT HAS MAGNITUDE RNORM, WITH
C           SIGN OPPOSITE TO THAT OF THE UNTRANSFORMED DIAGONAL ELEMENT.
C
            QRV(I,I) = -ESIGN*RNORM
C
C           DIVIDE ALL ELEMENTS OF THE HOUSEHOLDER VECTOR BY RNORM,
C           SO THAT THE ABSOLUTE VALUE OF THE I-TH COMPONENT IS EQUAL
C           TO THE NORMALIZING FACTOR FOR THE I-TH TRANSFORMATION.
```

```
C
         DO 520 K = NRNKP1, NCOL
            QRV(I, K) = QRV(I, K)/RNORM
  520    CONTINUE
C
C        -------------------------------------------------------------
C
C        APPLY THE TRANSFORMATION TO THE ROWS ABOVE ROW I.
C
C        -------------------------------------------------------------
C
         IF (I .EQ. 1) GO TO 560
         IM1 = I - 1
         DO 550 K = 1, IM1
C
C           COMPUTE THE DOT PRODUCT (DOTPRD) OF THE HOUSEHOLDER VECTOR
C           AND THE ROW TO WHICH THE TRANSFORMATION IS BEING APPLIED.
C           THE FIRST ELEMENT OF THE HOUSEHOLDER VECTOR IS TREATED
C           SEPARATELY BECAUSE IT IS STORED IN VALPHA(I).
C
            DOTPRD = VALPHA(I) * QRV(K,I)
            DO 530 J = NRNKP1, NCOL
               DOTPRD = DOTPRD + QRV(I,J) * QRV(K,J)
  530       CONTINUE
C
C           ----------------------------------------------------------
C
C           APPLY THE HOUSEHOLDER TRANSFORMATION TO THE ROW.  THIS IS
C           EQUIVALENT TO SUBTRACTING A MULTIPLE (GAMMA) OF THE
C           HOUSEHOLDER VECTOR FROM THE ROW. ONCE AGAIN, THE FIRST
C           ELEMENT OF THE HOUSEHOLDER VECTOR IS TREATED SEPARATELY.
C
C           ----------------------------------------------------------
C
            GAMMA = DOTPRD / DABS(VALPHA(I))
            QRV(K,I) = QRV(K,I) - GAMMA*VALPHA(I)
            DO 540 J = NRNKP1, NCOL
               QRV(K,J) = QRV(K,J) - GAMMA * QRV(I,J)
  540       CONTINUE
C
C        END LOOP OVER ROWS 1 THROUGH I-1.
C
  550    CONTINUE
C
C     END OF LOOP OVER THE TRANSFORMATIONS.
C
  560 CONTINUE
      IERROR = 0
      RETURN
      END
```

6.6. Subroutine LENSQ

6.6.1. Purpose

The subroutine LENSQ computes the squared Euclidean length of a vector in a manner designed to avoid underflow.

6.6.2. Method

The component of maximum magnitude in the vector is located in an initial scan. Thereafter, the squared length of a "normalized" vector is computed, where each component of the original vector is divided by the maximum; the squared length of the normalized vector must lie between unity and the number of components.

An additional protection against underflow is obtained by not including a component of the normalized vector in the calculation of the squared length if its ratio to the maximum is less than a small tolerance (10^{-20}).

The procedure followed in LENSQ is fairly simple, and is not the most efficient possible; a thorough treatment of this seemingly simple but surprisingly complex problem is given in Blue (1978).

6.6.3. Keywords

Euclidean length; squared length.

6.6.4. Source language

Fortran. The code in LENSQ has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.6.5. Specification and parameters

See accompanying listing.

6.6.6. Error indicators

See accompanying listing. If the squared length of the vector is too large or small to be represented on the machine, overflow or underflow will occur.

6.6.7. Auxiliary routines

LENSQ calls the standard function DABS.

6.6.8. Program size

22 Fortran source statements.

6.6.9. Array storage

No locally declared arrays.

6.6.10. Timing

The computation of the squared Euclidean length for a vector of length n requires, in general, $2n$ comparisons, n divisions, n additions, and $n + 2$ multiplications.

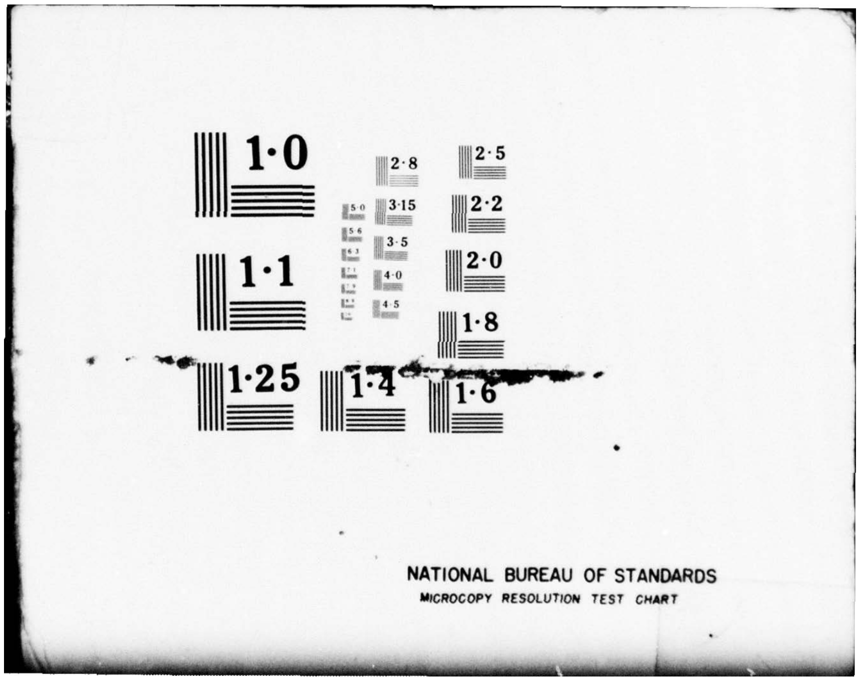
6.6.11. Further comments

None.

AD-A062 376 STANFORD UNIV CALIF SYSTEMS OPTIMIZATION LAB F/G 12/1
FORTRAN SUBROUTINES TO SOLVE THE LINEAR LEAST-SQUARES PROBLEM A--ETC(U)
APR 78 M H WRIGHT, S C GLASSMAN N00014-75-C-0267
UNCLASSIFIED SOL-78-8 NL

2 OF 2
ADA
062376

END
DATE
FILMED
3 -79
DDC


1·0
2·8
2·5
5·0
3·15
2·2
5·6
3·5
6·3
2·0
1·1
4·0
4·5
1·8
1·25
1·4
1·6
NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

```
      SUBROUTINE LENSQ ( LEN,  VEC,  VLENSQ )
      INTEGER   LEN
      DOUBLE PRECISION   VEC(LEN), VLENSQ
C
C
C-----------------------------------------------------------------------
C
C  THE SUBROUTINE LENSQ COMPUTES THE SQUARED EUCLIDEAN LENGTH OF
C  A VECTOR, USING SOME SAFEGUARDS TO PREVENT UNDERFLOW.  IN
C  PARTICULAR, THE ELEMENT OF MAXIMUM MODULUS IN THE VECTOR IS
C  LOCATED DURING AN INITIAL SCAN.  IF THIS ELEMENT IS NON-ZERO,
C  THE SQUARED LENGTH OF A NORMALIZED VECTOR (CONSTRUCTED BY DIVIDING
C  EACH ELEMENT OF THE ORIGINAL VECTOR BY THE MAXIMUM) IS COMPUTED.
C  THE SQUARED LENGTH OF THE NORMALIZED VECTOR MUST LIE BETWEEN UNITY
C  AND THE NUMBER OF COMPONENTS.  THE SQUARED LENGTH OF THE
C  ORIGINAL VECTOR IS THEN GIVEN BY THE PRODUCT OF THE SQUARE OF
C  THE MAXIMUM ELEMENT AND THE SQUARED LENGTH OF THE NORMALIZED VECTOR.
C
C
C  THE FORMAL PARAMETERS OF LENSQ ARE:
C
C      LEN ---
C              INTEGER, INPUT ONLY.
C              THE LENGTH OF THE VECTOR.
C
C      VEC ---
C              FLOATING POINT ARRAY, OF LENGTH 'LEN', INPUT ONLY.
C              THE VECTOR WHOSE SQUARED LENGTH IS TO BE COMPUTED.
C
C      VLENSQ ---
C              FLOATING POINT, OUTPUT ONLY.
C              THE COMPUTED SQUARED LENGTH OF THE ORIGINAL VECTOR.
C
C
C
C
C   *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                 SYSTEMS OPTIMIZATION LABORATORY
C                 DEPARTMENT OF OPERATIONS RESEARCH
C                 STANFORD UNIVERSITY
C                 STANFORD, CALIFORNIA 94305
C
C   *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C     DECLARATION OF LOCAL VARIABLES.
C
      INTEGER   I
```

```
C
C-----------------------------------------------------------------------
C
C     DECLARATION OF LOCAL VARIABLES.
C
      INTEGER  I
      DOUBLE PRECISION   ABSV, RATIO, TOL, TVMX, VMAX
C
C     DECLARATION OF STANDARD FUNCTIONS.
C
      DOUBLE PRECISION   DABS
C
C  THE VALUE 'TOL' IS USED IN A TEST TO AVOID UNDERFLOW IN
C  FORMING A QUOTIENT.
C

      DATA TOL / 1.0D-20 /
C
C-----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 10 FINDS THE ELEMENT OF MAXIMUM MODULUS IN
C  THE ARRAY VEC.
C
C-----------------------------------------------------------------------
C
      VMAX = 0.0D+0
      DO 10 I = 1, LEN
         ABSV  = DABS(VEC(I))
         IF (ABSV .GT. VMAX)  VMAX = ABSV
   10 CONTINUE
C
C-----------------------------------------------------------------------
C
C  COMPUTE THE SQUARED LENGTH OF THE NORMALIZED VECTOR.  A FURTHER
C  SAFEGUARD AGAINST UNDERFLOW IS INCLUDED BY TESTING WHETHER
C  THE RATIO OF ANY ELEMENT TO THE MAXIMUM IS VERY SMALL, IN
C  WHICH CASE IT IS NOT INCLUDED IN THE SUM.
C
C-----------------------------------------------------------------------
C
      VLENSQ = 0.0D+0
      IF (VMAX .EQ. 0.0D+0)  RETURN
      TVMX = TOL*VMAX
      DO 20 I = 1, LEN
         IF (DABS(VEC(I)) .LE. TVMX)  GO TO 20
         RATIO = VEC(I)/VMAX
         VLENSQ = VLENSQ + RATIO*RATIO
   20 CONTINUE
      VLENSQ = VMAX*VMAX*VLENSQ
      RETURN
      END
```

6.7. Subroutine MNLNLS

6.7.1. Purpose

The subroutine MNLNLS is designed to compute the minimum-length least-squares solution for a given matrix and a single right-hand side. It also returns the residual vector, the transformed right-hand side, and an estimate of the rank.

6.7.2. Description of method

The solution procedure is described in Section 3.3. MNLNLS simply calls the appropriate subroutines to carry out each of the necessary computations.

6.7.3. Keywords

Linear least-squares; linear equations; minimum-length least-squares solution; overdetermined linear system; underdetermined linear system.

6.7.4. Source language

Fortran. The code in MNLNLS has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.7.5. Specification and parameters

See accompanying listing.

6.7.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.7.7. Auxiliary routines

MNLNLS calls the subroutines HREDL, HREDR, and QRVSLV, which in turn call HMULTC, LENSQ, QMULVC, VMULVC, TRSLV, and UNSCRM.

6.7.8. Program size

16 Fortran source statements.

6.7.9. Array storage

No internally declared arrays.

6.7.10. Timing

The number of arithmetic operations is the sum of the operations required by HREDL, HREDR, and QRVSLV.

6.7.11. Further comments

A column interchange strategy is used during the triangular reduction from the left, and the "size" of a remaining column is considered to be the ratio of its current length to its original length.

If the least-squares problem is to be solved for one matrix and several right-hand sides, the user should:

(1) call MNLNLS only once, for the first right-hand side. Following execution of MNLNLS, the matrix A will have been transformed, and information about the transformations will be stored in A, and in the vectors ITEMP, TEMP2, and TEMP3;

(2) for each subsequent right-hand side, say the vector C, call the subroutine QRVSLV as follows:

```
QRVSLV(NROW, NCOL, NMAX, NDIM, A, .TRUE., .TRUE., NRANK, ITEMP,
TEMP2, TEMP3, C, XC, RESC, LERROR),
```

where XC will contain the solution and RESC will contain the residual vector corresponding to C. All the parameters through TEMP3 must be exactly as given above, so that the correct information is available in QRVSLV.

```
      SUBROUTINE MNLNLS ( NROW, NCOL, NMAX, NDIM, A, COLEPS, B,
     1   NRANK, X, ITEMP, TEMP1, TEMP2, TEMP3, IERROR )
C
      INTEGER   NROW, NCOL, NMAX, NDIM, NRANK, IERROR
      INTEGER   ITEMP(NCOL)
      DOUBLE PRECISION   COLEPS
      DOUBLE PRECISION   A(NDIM,NCOL), B(NROW), X(NCOL), TEMP1(NMAX),
     1     TEMP2(NCOL), TEMP3(NCOL)
C
C
C-----------------------------------------------------------------------
C
C     THE SUBROUTINE MNLNLS COMPUTES THE MINIMUM-LENGTH LEAST-
C   SQUARES SOLUTION OF A LINEAR SYSTEM, AND CAN THEREFORE BE
C   USED TO SOLVE NON-SINGULAR, OVER-DETERMINED, AND UNDER-
C   DETERMINED LINEAR SYSTEMS.
C
C      THE PROBLEM TO BE SOLVED BY MNLNLS IS:  GIVEN A REAL
C   NROW BY NCOL MATRIX  A , AND AN NROW-VECTOR  B , FIND THE
C   NCOL-VECTOR  X  OF MINIMUM EUCLIDEAN LENGTH SUCH THAT THE
C   EUCLIDEAN NORM OF THE RESIDUAL VECTOR  (A*X - B) IS A MINIMUM.
C
C      IN ORDER TO SOLVE THIS PROBLEM, MNLNLS CALLS A SET
C   OF SUBROUTINES THAT REDUCE THE MATRIX  A  TO UPPER
C   TRIANGULAR FORM, AND THEN SOLVE A TRANSFORMED PROBLEM.
C   FULL DETAILS OF THE SOLUTION PROCEDURE ARE GIVEN IN THE
C   EXTERNAL DOCUMENTATION.  THE ROUTINES REQUIRED BY MNLNLS ARE
C   HMULTC, HREDL, HREDR, LENSQ, QMULVC, QRVSLV, TRSLV, UNSCRM,
C   AND VMULVC.
C
C
C   THE FORMAL PARAMETERS OF MNLNLS ARE:
C
C   NROW --
C        INTEGER, INPUT ONLY.
C        THE NUMBER OF ROWS IN THE MATRIX  A.  MUST BE .GT. 0.
C
C   NCOL --
C        INTEGER, INPUT ONLY.
C        THE NUMBER OF COLUMNS IN THE MATRIX  A.  MUST BE .GT. 0.
C
C   NMAX --
C        INTEGER, INPUT ONLY.
C        THE MAXIMUM OF (NROW, NCOL).  THIS VALUE IS NEEDED FOR
C        FORTRAN TO ALLOW THE CORRECT DYNAMIC DIMENSION OF THE
C        TEMPORARY ARRAY TEMP1.
C
C   NDIM --
C        INTEGER, INPUT ONLY.
C        THE DECLARED ROW DIMENSION OF THE MATRIX  A.  MUST BE .GE.
C        NROW.
C
```

```
C  A --
C       DOUBLE PRECISION ARRAY, OF CONCEPTUAL DIMENSION NROW BY
C       NCOL, AND DECLARED DIMENSION NDIM BY NCOL.  INPUT AND OUTPUT.
C       ON INPUT, THE ARRAY  A  SHOULD CONTAIN THE  MATRIX INVOLVED
C       IN THE LEAST-SQUARES PROBLEM.  ON OUTPUT, THE MATRIX  A  HAS
C       BEEN TRANSFORMED, AND CONTAINS INFORMATION ABOUT THE REDUCED
C       FORM AND THE TRANSFORMATIONS USED IN THE REDUCTION.  FOR
C       DETAILS, THE USER SHOULD REFER TO EXTERNAL DOCUMENTATION.
C
C  COLEPS --
C       DOUBLE PRECISION, INPUT ONLY.
C       USUALLY, COLEPS SHOULD BE SET TO THE MAXIMUM OF THE SQUARE
C       ROOT OF MACHINE PRECISION, AND THE RELATIVE ACCURACY OF THE
C       ELEMENTS IN THE MATRIX  A  AND THE VECTOR  B.  OTHER VALUES
C       MAY BE USED, BUT THE USER SHOULD READ THE DOCUMENTATION
C       FOR THE ROUTINE HREDL BEFORE DOING SO.
C
C  B --
C       DOUBLE PRECISION VECTOR, OF LENGTH NROW.  INPUT AND OUTPUT.
C       ON INPUT, B SHOULD CONTAIN THE RIGHT-HAND SIDE OF THE
C       LEAST-SQUARES PROBLEM.  ON OUTPUT, THE VECTOR  B  WILL
C       CONTAIN THE TRANSFORMED RIGHT-HAND SIDE, Q*B.
C
C  NRANK --
C       INTEGER, OUTPUT ONLY.
C       THE ESTIMATED NUMERICAL RANK OF THE MATRIX  A.
C
C  X --
C       DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C       THE COMPUTED MINIMUM-LENGTH LEAST-SQUARES SOLUTION.
C
C  ITEMP --
C       INTEGER ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C       ITEMP WILL CONTAIN A RECORD OF THE COLUMN INTERCHANGES
C       CARRIED OUT DURING THE TRIANGULAR REDUCTION.  IT
C       CORRESPONDS TO THE PARAMETER 'IPERM' IN THE ROUTINE
C       HREDL.
C
C  TEMP1 --
C       DOUBLE PRECISION ARRAY, OF LENGTH NMAX.  OUTPUT ONLY.
C       ON EXIT, TEMP1 WILL CONTAIN THE RESIDUAL VECTOR FOR THE
C       LEAST-SQUARES PROBLEM.
C
C  TEMP2 --
C       DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C       ON EXIT, TEMP2 WILL CONTAIN INFORMATION ABOUT THE
C       TRANFORMATIONS USED DURING THE REDUCTION FROM THE LEFT.
C       TEMP2 CORRESPONDS TO THE ARRAY 'HVECL' IN THE ROUTINE
C       HREDL.
C
C  TEMP3 --
C       DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C       ON EXIT, TEMP3 WILL CONTAIN INFORMATION ABOUT THE
```

```
C       TRANSFORMATIONS USED DURING THE POSSIBLE REDUCTION
C       FROM THE RIGHT.  TEMP3 CORRESPONDS TO THE ARRAY 'HVECR'
C       IN THE ROUTINE HREDR.
C
C  IERROR --
C       INTEGER, OUTPUT ONLY.
C       AN ERROR FLAG, WITH THE FOLLOWING POSSIBLE VALUES.
C       IERROR = 0 : NO ERRORS, NORMAL TERMINATION.
C       IERROR > 0 : ERROR IN HREDL (CHECK DOCUMENATION OF HREDL).
C
C
C  *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                SYSTEMS OPTIMIZATION LABORATORY
C                DEPARTMENT OF OPERATIONS RESEARCH
C                STANFORD UNIVERSITY
C                STANFORD, CALIFORNIA 94305
C
C  *** DATE:     MARCH 1978
C
C-----------------------------------------------------------------------
C
C     DECLARATION OF LOCAL VARIABLES.
C
      LOGICAL INTERC, MINLEN
      INTEGER MODTOL
C
C
      INTERC = .TRUE.
      MODTOL = 1
C
C     HREDL CARRIES OUT THE REDUCTION OF  A  TO UPPER TRAPEZOIDAL
C     FORM, AND COMPUTES THE ESTIMATED RANK OF   A.
C
      CALL HREDL(NROW, NCOL, NDIM, A, COLEPS, MODTOL, INTERC, TEMP1,
     1  NRANK, TEMP2, ITEMP, TEMP3, IERROR)
      IF (IERROR .NE. 0)  RETURN
C
C     IF NECESSARY, HREDR REDUCES THE UPPER TRAPEZOID OF THE
C     TRANSFORMED MATRIX  A  TO AN UPPER TRIANGLE FROM THE RIGHT.
C
      IF (NRANK .LT. NCOL)  CALL HREDR(NRANK, NCOL, NDIM, A, TEMP3,
     1  IERROR)
      MINLEN = .TRUE.
C
C     QRVSLV COMPLETES THE SOLUTION OF THE LEAST-SQUARES
C     PROBLEM BY TRANSFORMING THE RIGHT-HAND SIDE, SOLVING A
C     TRIANGULAR SYSTEM, AND, IF NECESSARY, TRANSFORMING THE
C     SOLUTION.
C
      CALL QRVSLV(NROW, NCOL, NMAX, NDIM, A, INTERC, MINLEN, NRANK,
     1  ITEMP, TEMP2, TEMP3, B, X, TEMP1, IERROR)
      RETURN
      END
```

6.8. Subroutine QMULVC

6.8.1. Purpose

The subroutine QMULVC applies to a vector the special sequence of r Householder tranformations constructed by HREDL in reducing an m by n matrix of rank r to upper trapezoidal form from the left. Its principal uses are in transforming the right-hand side of a linear least-squares problem, and in back-transforming the residual vector.

The sequence of transformations, which are stored in compact form (see Section 6.3.11), can be applied in either forward or backward order. Let the set of transformations applied on the left be:

$$H_r \cdots H_1 \equiv Q \quad ,$$

where H_j reduces the j-th column to the appropriate form. QMULVC can apply the transformations in forward order (multiply by Q), or in reverse order (multiply by Q^T).

6.8.2. Description of method

The sequence of transformations is assumed to be represented in the compact, normalized form described in Section 6.3.11 and 6.4.11. The transformations are applied in the desired order, taking advantage of the known structure of the Householder vectors (i.e., H_k does not alter components 1 through $k - 1$).

6.8.3. Keywords

Householder transformation.

6.8.4. Source language

Fortran. The code in QMULVC has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.8.5. Specification and parameters

See accompanying listing.

6.8.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.8.7. Auxiliary routines

QMULVC calls the standard functions IABS and DABS.

6.8.8. Program size

38 Fortran source statements.

6.8.9. Array storage

No locally declared arrays.

6.8.10. Timing

The number of arithmetic operations required by QMULVC is in general of approximate order $2mr - r^2$.

6.8.11. Further comments

The vectors that define the Householder-transformations must be stored and normalized as described in Sections 6.3.11 and 6.4.11. If HVECL(k) = 0, the k-th transformation is skipped (the vector is not altered by it).

```
      SUBROUTINE QMULVC ( NROW, NRANK, NDIM, QRV, HVECL, VECIN,
     1   MULINC, VECOUT, IERROR )
C
      INTEGER    NROW, NRANK, NDIM, MULINC, IERROR
      DOUBLE PRECISION    QRV(NDIM, NRANK), HVECL(NRANK), VECIN(NROW),
     1    VECOUT(NROW)
C
C
C-----------------------------------------------------------------------
C
C   THE SUBROUTINE QMULVC APPLIES A SEQUENCE OF HOUSEHOLDER
C   TRANSFORMATIONS TO A VECTOR, EITHER IN FORWARD OR BACKWARD
C   ORDER.  THE SEQUENCE OF TRANSFORMATIONS IS ASSUMED TO HAVE BEEN
C   GENERATED BY THE SUBROUTINE HREDL, AND FULL DETAILS ABOUT THE
C   PROCEDURE ARE GIVEN IN THE EXTERNAL DOCUMENTATION AND IN THE
C   COMMENTS AT THE BEGINNING OF HREDL.
C
C   THE FORMAL PARAMETERS OF QMULVC ARE --
C
C   NROW -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF COMPONENTS IN THE INPUT AND OUTPUT VECTORS, AND
C         THE NUMBER OF ROWS OF QRV.
C
C   NRANK -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF HOUSEHOLDER TRANSFORMATIONS TO BE APPLIED.
C
C   NDIM -
C         INTEGER, INPUT ONLY.
C         THE DECLARED ROW DIMENSION OF THE MATRIX QRV.  MUST BE .GE.
C         NROW.
C
C   QRV -
C         DOUBLE PRECISION ARRAY, OF DECLARED ROW DIMENSION NDIM, WITH
C         AT LEAST NRANK COLUMNS.  INPUT ONLY.
C         THE MATRIX QRV MUST CONTAIN THE OUTPUT OF THE SUBROUTINE HREDL.
C
C   HVECL -
C         DOUBLE PRECISION ARRAY, OF LENGTH NRANK.  INPUT ONLY.
C         THE VECTOR HVECL MUST HAVE BEEN GENERATED BY SUBROUTINE HREDL.
C
C   VECIN -
C         DOUBLE PRECISION ARRAY, OF LENGTH NROW.  INPUT ONLY.
C         THE VECTOR TO BE TRANSFORMED.
C
C   MULINC -
C         INTEGER, INPUT ONLY.
C         THE VALUE OF MULINC INDICATES THE ORDER IN WHICH THE
C         TRANSFORMATIONS ARE TO BE APPLIED.  THE VECTOR U(I)
```

```
C        CORRESPONDING TO THE I-TH TRANSFORMATION P(I) HAS ZEROS IN
C        COMPONENTS 1 THROUGH I-1, AND NON-ZEROS ELSEWHERE.  LET
C        Q DENOTE THE PRODUCT P(NRANK) * ... * P(2) * P(1).  IF
C        MULINC = 1, THE TRANSFORMATIONS ARE APPLIED IN FORWARD ORDER,
C        I.E.,
C
C           P(NRANK) * ... * P(2) * P(1) * VECIN = VECOUT, OR,
C
C           Q * VECIN = VECOUT.
C
C        IF MULINC = -1, THE TRANSFORMATIONS ARE APPLIED IN THE REVERSE
C        ORDER, I.E.,
C
C           P(1) * P(2) * ... * P(NRANK) * VECIN = VECOUT, OR,
C
C           (Q TRANSPOSE) * VECIN = VECOUT.
C
C
C  VECOUT -
C       DOUBLE PRECISION ARRAY, OF LENGTH NROW.  OUTPUT ONLY.
C       THE TRANSFORMED VECTOR.  THE ACTUAL PARAMETERS CORRESPONDING TO
C       VECIN AND VECOUT MAY BE THE SAME.
C
C  IERROR -
C       INTEGER, OUTPUT ONLY.
C       AN ERROR FLAG, WITH THE FOLLOWING POSSIBLE VALUES.
C          IERROR = 0 - NO ERRORS, NORMAL TERMINATION.
C          IERROR = 1 - INVALID INPUT PARAMETER.
C
C
C
C   *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                 SYSTEMS OPTIMIZATION LABORATORY
C                 DEPARTMENT OF OPERATIONS RESEARCH
C                 STANFORD UNIVERSITY
C                 STANFORD, CALIFORNIA 94305
C
C   *** DATE:     DECMEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C       DECLARATION OF LOCAL VARIABLES.
C
        INTEGER   I, JH, JHP1, NICOP
        DOUBLE PRECISION   DOTPRD, FACT
C
C       DECLARATION OF STANDARD FUNCTIONS.
C
```

```
      INTEGER   IABS
      DOUBLE PRECISION   DABS
C
C
C
C  CHECK FOR ERROR IN INPUT PARAMETERS.
C
      IERROR = 1
      IF (IABS(MULINC) .NE. 1 .OR. NROW .LE. 0 .OR. NRANK .EQ. 0
     1    .OR. NRANK .GT. NROW)  RETURN
C
C-----------------------------------------------------------------------
C
C  SET UP THE CONTROLLING INDICES FOR EITHER FORWARD OR BACKWARD
C  APPLICATION OF THE TRANSFORMATIONS.
C
C-----------------------------------------------------------------------
C
      IF (MULINC .LT. 0)  GO TO 10
      JH = 1
      GO TO 20
   10 JH = NRANK
   20 CONTINUE
      DO 30 I = 1, NROW
         VECOUT(I) = VECIN(I)
   30 CONTINUE
      IERROR = 2
C
C-----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 100 IS OVER THE NUMBER OF TRANSFORMATIONS
C  TO BE APPLIED.  THE INDEX JH IS THE INDEX OF THE TRANSFORMATION
C  APPLIED AT THE CURRENT STEP.  JH RUNS FROM 1 TO NRANK IF MULINC = 1
C  (FORWARD APPLICATION) OR FROM NRANK TO 1 IF MULINC = -1 (BACKWARD
C  APPLICATION).
C
C-----------------------------------------------------------------------
C
C
      DO 100 NLOOP = 1, NRANK
C
C        TEST WHETHER THE JH-TH TRANSFORMATION IS TO BE SKIPPED.
C
         IF (HVECL(JH) .EQ. 0.0D+0)  GO TO 70
C
C        FORM THE NORMALIZED INNER PRODUCT OF THE JH-TH HOUSEHOLDER
C        TRANSFORMATION AND THE TRANSFORMED VECTOR.  THE JH-TH
C        COMPONENT OF THE HOUSEHOLDER VECTOR IS STORED IN HVECL(JH),
C        AND THE FIRST (JH-1) COMPONENTS ARE ZERO.
C
```

```
         DOTPRD = HVECL(JH)*VECOUT(JH)
         IF (JH .EQ. NROW)  GO TO 50
         JHP1 = JH + 1
         DO 40 I = JHP1, NROW
            DOTPRD = DOTPRD + VECOUT(I)*QRV(I,JH)
   40    CONTINUE
   50    CONTINUE
C
C        APPLY THE TRANSFORMATION BY SUBTRACTING AN APPROPRIATE
C        MULTIPLE OF THE HOUSEHOLDER VECTOR, WHERE THE JH-TH
C        COMPONENT IS TREATED SEPARATELY.
C
         FACT = DOTPRD/DABS(HVECL(JH))
         VECOUT(JH) = VECOUT(JH) - FACT*HVECL(JH)
         IF (JH .EQ. NROW)  GO TO 70
         DO 60 I = JHP1, NROW
            VECOUT(I) = VECOUT(I) - FACT*QRV(I,JH)
   60    CONTINUE
   70    JH = JH + MULINC
  100 CONTINUE
C
C  THE LOOP HAS TERMINATED NORMALLY.
C
      LERROR = 0
      RETURN
      END
```

6.9. Subroutine QRVSLV

6.9.1. Purpose

The subroutine QRVSLV is used to solve the linear least-squares problem for a single right-hand side after the matrix involved has been reduced to an appropriate form through application of Householder transformations by subroutines HREDL and HREDR.

6.9.2. Descruption of method

See Section 3.3.

6.9.3. Keywords

Linear least-squares; linear equations; overdetermined linear system; underdetermined linear system.

6.9.4. Source language

Fortran. The code in QRVSLV has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.9.5. Specification and parameters

See accompanying listing.

6.9.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.9.7. Auxiliary routines

QRVSLV calls the subroutines QMULVC, TRSLV, UNSCRM, and VMULVC.

6.9.8. Program size

38 Fortran source statements

6.9.9. Array storage

No locally declared arrays.

6.9.10. Timing

The number of arithmetic operations is the sum of the operations required for QMULVC, TRSLV, UNSCRM, and VMULVC.

6.9.11. Further comments

None.

```
      SUBROUTINE QRVSLV ( NROW, NCOL, NMAX, NDIM, QRV, INTERC,
     1  MINLEN, NRANK, IPERM, HVECL, HVECR, B, X, RES, LERROR )
C
      INTEGER NROW, NCOL, NMAX, NDIM, NRANK, LERROR
      INTEGER IPERM(NCOL)
      LOGICAL INTERC, MINLEN
      DOUBLE PRECISION QRV(NDIM, NCOL), HVECL(NCOL), HVECR(NCOL),
     1  B(NROW), X(NCOL), RES(NMAX)
C
C
C-----------------------------------------------------------------------
C
C
C        THE SUBROUTINE QRVSLV IS USED TO SOLVE THE LINEAR LEAST-
C     SQUARES PROBLEM, MIN 2-NORM(QRV*X - B)**2.  IT IS ASSUMED
C     THAT THE MATRIX QRV HAS BEEN REDUCED TO UPPER TRIANGULAR FORM
C     BY APPLICATION OF HOUSEHOLDER TRANSFORMATIONS FROM THE LEFT,
C     POSSIBLY WITH COLUMN INTERCHANGES, AND (OPTIONALLY) THAT THE
C     MATRIX WAS FURTHER REDUCED BY APPLICATION OF HOUSEHOLDER
C     TRANSFORMATIONS ON THE RIGHT IF THE MATRIX APPEARED TO BE RANK-
C     DEFICIENT.  THESE CALCULATIONS ARE CARRIED OUT BY THE SUBROUTINES
C     HREDL AND HREDR, AND OTHER AUXILIARY ROUTINES (DESCRIBED IN DETAIL
C     IN THE DOCUMENTATION FOR HREDL AND HREDR).
C
C        IN ORDER TO SOLVE THE LEAST-SQUARES PROBLEM FOR A GIVEN RIGHT-
C     HAND SIDE VECTOR B, THE FOLLOWING STEPS MUST BE CARRIED OUT:
C
C         (A) APPLY TO B THE TRANSFORMATIONS THAT WERE APPLIED ON
C         THE LEFT TO QRV DURING THE EARLIER REDUCTION, YIELDING
C         A TRANSFORMED VECTOR Q*B.
C
C         (B) SOLVE THE LINEAR SYSTEM R*Y = BB, WHERE R IS THE NRANK
C         BY NRANK UPPER TRIANGULAR MATRIX IN THE UPPER LEFT CORNER
C         OF THE TRANSFORMED QRV, AND BB CONTAINS THE FIRST NRANK
C         COMPONENTS OF THE VECTOR Q*B.  NRANK IS THE NUMERICAL RANK
C         OF QRV, ESTIMATED DURING THE EARLIER REDUCTION.
C
C         (C) IF ANY TRANSFORMATIONS WERE APPLIED TO QRV ON THE RIGHT,
C         APPLY THESE TRANSFORMATIONS TO Y, YIELDING V*Y = X.
C         IF NO TRANSFORMATIONS WERE APPLIED ON THE RIGHT, X = Y.
C
C         (D) IF COLUMN INTERCHANGES WERE CARRIED OUT DURING THE
C         REDUCTION OF QRV, INTERCHANGE THE APPROPRIATE COMPONENTS
C         OF X.
C
C         (E) IN ORDER TO OBTAIN THE RESIDUAL VECTOR FOR THE ORIGINAL
C         PROBLEM, APPLY Q TRANSPOSE TO THE RESIDUAL OF THE TRANS-
C         FORMED PROBLEM (I.E., APPLY THE HOUSEHOLDER TRANSFORMATIONS
C         IN REVERSE ORDER).  BY CONSTRUCTION, THE RESIDUAL OF THE
C         TRANSFORMED PROBLEM IS ZERO IN COMPONENTS 1 TO NRANK, AND
```

```
C        THE REMAINING COMPONENTS ARE EQUAL TO THE CORRESPONDING
C        COMPONENTS OF Q*B.
C
C
C     THE FORMAL PARAMETERS OF QRVSLV ARE:
C
C     NROW -
C          INTEGER, INPUT ONLY.
C          THE NUMBER OF ROWS OF QRV, AND THE LENGTH OF THE VECTOR B.
C
C     NCOL -
C          INTEGER, INPUT ONLY.
C          THE NUMBER OF COLUMNS OF QRV, AND THE LENGTH OF THE VECTORS
C          IPERM, HVECL, HVECR, AND X.
C
C     NMAX -
C          INTEGER, INPUT ONLY.
C          NMAX = MAX (NROW, NCOL).  THE LENGTH OF THE VECTOR RES.
C
C     NDIM -
C          INTEGER, INPUT ONLY.
C          THE DECLARED ROW DIMENSION OF THE MATRIX QRV.  MUST BE
C          .GE. NROW.
C
C     QRV -
C         DOUBLE PRECISION ARRAY, OF DECLARED ROW DIMENSION NDIM, AND
C         CONCEPTUAL SIZE NROW BY NCOL.  INPUT ONLY.
C         QRV CONTAINS THE RESULTS OF COMPUTATION OF HREDL AND HREDR,
C         AND CORRESPONDS TO THE ORIGINAL MATRIX IN THE LEAST-SQUARES
C         PROBLEM.
C
C     INTERC -
C         LOGICAL, INPUT ONLY.
C         IF INTERC IS .TRUE., COLUMN INTERCHAMNGES MAY HAVE BEEN
C         CARRIED OUT DURING THE REDUCTION FROM THE LEFT. THE VALUE
C         OF INTERC IN THE CALL TO QRVSLV SHOULD BE THE SAME AS THE
C         VALUE OF INTERC IN THE EARLIER CALL TO HREDL.
C
C     MINLEN -
C         LOGICAL, INPUT ONLY.
C         MINLEN SHOULD BE SET TO .TRUE. IF THE MINIMUM-LENGTH LEAST-
C         SQUARES SOLUTION IS DESIRED, AND TO .FALSE. OTHERWISE.
C         IF MINLEN IS .TRUE., INTERC SHOULD HAVE BEEN .TRUE.
C
C     NRANK -
C         INTEGER, INPUT ONLY.
C         THE ESTIMATED RANK OF QRV, AS COMPUTED BY THE ROUTINE HREDL.
C
C     IPERM -
C         INTEGER ARRAY, OF LENGTH NCOL.
```

```
C         IPERM CONTAINS INFORMATION ABOUT THE COLUMN INTERCHANGES
C         THAT MAY HAVE BEEN CARRIED OUT BY HREDL.
C
C     HVECL -
C         DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  INPUT ONLY.
C         THE VECTOR HVECL IS GENERATED BY THE SUBROUTINE HREDL, AND
C         CONTAINS INFORMATION ABOUT THE TRANSFORMATIONS ON THE LEFT.
C
C     HVECR -
C         DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  INPUT ONLY.
C         THE VECTOR HVECR IS GENERATED BY THE SUBROUTINE HREDR, AND
C         CONTAINS INFORMATION ABOUT THE TRANSFORMATIONS ON THE
C         RIGHT.
C
C     B -
C         DOUBLE PRECISION VECTOR, OF LENGTH NROW.  INPUT AND OUTPUT.
C         ON ENTRY TO QRVSLV, B SHOULD CONTAIN THE RIGHT-HAND SIDE
C         OF THE LEAST-SQUARES PROBLEM.  ON EXIT FROM QRVSLV,
C         B HAS BEEN OVERWRITTEN BY Q*B, WHERE Q IS THE PRODUCT OF
C         THE HOUSEHOLDER TRANSFORMATIONS APPLIED ON THE LEFT.
C
C     X -
C         DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C         X WILL CONTAIN THE COMPUTED SOLUTION OF THE LEAST-SQUARES
C         PROBLEM.
C
C     RES -
C         DOUBLE PRECISION ARRAY, OF LENGTH NMAX.  OUTPUT ONLY.
C         RES WILL CONTAIN THE RESIDUAL VECTOR OF THE ORIGINAL
C         LEAST-SQUARES PROBLEM.
C
C     LERROR -
C         INTEGER, OUTPUT ONLY.
C         AN ERROR FLAG, WITH THE FOLLOWING POSSIBLE VALUES.
C            LERROR = 0 - NO ERRORS, NORMAL TERMINATION.
C            LERROR < 10 - ERROR IN QMULVC (SEE QMULVC DOCUMENTATION).
C            LERROR < 20 - ERROR IN TRSLV (SEE TRSLV DOCUMENTATION).
C            LERROR < 30 - ERROR IN VMULVC (SEE VMULVC DOCUMENTATION).
C
C
C     THE SUBROUTINE QRVSLV CALLS THE AUXILIARY SUBROUTINES QMULVC,
C   TRSLV, VMULVC, AND UNSCRM.
C
C
C *** AUTHORS:   MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                SYSTEMS OPTIMIZATION LABORATORY
C                DEPARTMENT OF OPERATIONS RESEARCH
C                STANFORD UNIVERSITY
C                STANFORD, CALIFORNIA 94305
C
```

```
C   *** DATE:      DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C   DECLARATION OF LOCAL VARIABLES.
C
      INTEGER I, NRNKP1
C
C
C     APPLY Q TO THE RIGHT-HAND SIDE VECTOR, B.
C
      CALL QMULVC(NROW, NRANK, NDIM, QRV, HVECL, B, 1, B, LERROR)
C
      IF (LERROR .NE. 0)  RETURN
C
C    SOLVE THE UPPER TRIANGULAR SYSTEM, WITH THE FIRST NRANK
C    COMPONENTS OF THE TRANSFORMED B AS THE RIGHT-HAND SIDE.
C
      CALL TRSLV(NRANK, NCOL, NDIM, QRV, B, -1, RES, LERROR)
C
      IF (LERROR .EQ. 0)  GO TO 20
      LERROR = LERROR + 10
      RETURN
C
   20 CONTINUE
      IF (.NOT. INTERC .OR. .NOT. MINLEN .OR. NRANK .EQ. NCOL)
     1   GO TO 30
C
C     APPLY THE MATRIX V TO THE VECTOR RES, WHICH CONTAINS THE
C     SOLUTION OF THE LINEAR SYSTEM.
C
      CALL VMULVC(NCOL, NRANK, NDIM, QRV, HVECR, RES, 1, RES, LERROR)
      IF (LERROR .EQ. 0)  GO TO 30
      LERROR = LERROR + 20
      RETURN
   30 CONTINUE
      IF (INTERC)  GO TO 50
C
C     IF THERE WERE NO INTERCHANGES, STORE THE SOLUTION IN X.
C
      DO 40 I = 1, NCOL
         X(I) = RES(I)
   40 CONTINUE
      GO TO 60
C
C     IF COLUMN INTERCHANGES WERE MADE, RE-ORDER THE SOLUTION VECTOR.
C
   50 CONTINUE
      CALL UNSCRM(1, NCOL, IPERM, RES, X)
   60 CONTINUE
C
C     STORE THE RESIDUAL VECTOR OF THE TRANSFORMED PROBLEM IN THE VECTO
C     RES.  THE FIRST NRANK COMPONENTS OF THE TRANSFORMED RESIDUAL ARE
C     ASSUMED TO BE EXACTLY ZERO, AND THE LAST (NROW-NRANK) COMPONENTS
C     ARE GIVEN BY THE LAST (NROW-NRANK) COMPONENTS OF THE TRANSFORMED
C     RIGHT-HAND SIDE.
C
      DO 70 I = 1, NRANK
         RES(I) = 0.0D+0
   70 CONTINUE
      IF (NRANK .EQ. NROW)  RETURN
      NRNKP1 = NRANK + 1
      DO 80 I = NRNKP1, NROW
         RES(I) = B(I)
   80 CONTINUE
C
C     APPLY THE HOUSEHOLDER TRANSFORMATIONS IN REVERSE ORDER, TO OBTAIN
C     THE RESIDUAL VECTOR OF THE ORIGINAL PROBLEM.
C
      CALL QMULVC(NROW, NRANK, NDIM, QRV, HVECL, RES, -1, RES, LERROR)
      RETURN
      END
```

6.10. Subroutine TRSLV

6.10.1. Purpose

Subroutine TRSLV returns the solution of a non-singular r by r triangular linear system, where the triangular matrix is assumed to be stored in the first r columns of an r by n matrix. In the current package, TRSLV is called by QRVSLV in solving the linear least-squares problem.

6.10.2. Description of method

If the matrix is lower triangular, forward elimination is used; if the matrix is upper triangular, the solution is obtained by back-substitution.

6.10.3. Keywords

Triangular linear system; forward elimination; back-substitution.

6.10.4. Source language

Fortran. The code in TRSLV has been checked by the PFORT verifier, and is WATFIV-compatible. All variables are explicitly declared.

6.10.5. Specification and parameters

See accompanying listing.

6.10.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.10.7. Auxiliary routines

None.

6.10.8. Program size

35 Fortran source statements.

6.10.9. Array storage

No locally declared arrays.

6.10.10. Timing

The number of arithmetic operations required to solve an r by r triangular system is of order $r^2/2$.

6.10.11. Further comments

Because TRSLV may be called to solve an upper trapezoidal system involving an r by n $(n \geq r)$ matrix, the number of components in the solution is allowed to exceed r; components $r + 1$ through n of the solution are simply set to zero.

```
      SUBROUTINE TRSLV ( N, NCOL, NDIM, QRV, B, INCSL, Y, LERROR )
C
      INTEGER   N, NCOL, NDIM, INCSL, LERROR
      DOUBLE PRECISION   QRV(NDIM, N), B(N), Y(NCOL)
C
C
C-----------------------------------------------------------------------
C
C  THE SUBROUTINE TRSLV IS USED TO SOLVE A TRIANGULAR SYSTEM OF LINEAR
C  EQUATIONS (EITHER LOWER OR UPPER), WHERE THE TRIANGULAR MATRIX IS
C  IN THE UPPER LEFT N BY N CORNER OF QRV.  TRSLV IS
C  SPECIFICALLY DESIGNED TO BE USED IN SOLVING THE LINEAR LEAST-
C  SQUARES PROBLEM.
C
C  THE FORMAL PARAMETERS OF TRSLV ARE --
C
C  N -
C        INTEGER, INPUT ONLY.
C        THE SIZE OF THE TRIANGULAR MATRIX.
C
C  NCOL -
C        INTEGER, INPUT ONLY.
C        THE NUMBER OF UNKNOWNS, AND THE NUMBER OF COLUMNS IN QRV.
C        MUST BE .GE. N.
C
C  NDIM -
C        INTEGER, INPUT ONLY.
C        THE DECLARED ROW DIMENSION OF QRV.  MUST BE .GE. N.
C
C  QRV -
C        DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NDIM BY NCOL.
C        THE TRIANGULAR MATRIX USED IN TRSLV IS CONTAINED IN THE UPPER
C        LEFT CORNER OF QRV.
C
C  B -
C        DOUBLE PRECISION VECTOR, OF LENGTH N.  INPUT ONLY.
C        THE RIGHT-HAND SIDE OF THE SYSTEM OF EQUATIONS.
C
C  INCSL -
C        INTEGER, INPUT ONLY.
C        THE VALUE OF INCSL INDICATES WHETHER THE MATRIX IS LOWER OR
C        UPPER TRIANGULAR.  IF INCSL = 1, THE MATRIX IS CONSIDERED TO BE
C        LOWER TRIANGULAR, AND FORWARD ELIMINATION IS CARRIED OUT.
C        IF INCSL = -1, THE MATRIX IS UPPER TRIANGULAR, AND BACKWARD
C        SOLUTION IS USED.
C
C  Y -
C        DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C        THE FIRST N COMPONENTS OF Y ARE THE SOLUTION OF THE
C        TRIANGULAR SYSTEM, AND THE REMAINING COMPONENTS ARE SET TO
```

```
C          ZERO.
C
C   LERROR -
C          INTEGER, OUTPUT ONLY.
C          AN ERROR FLAG, WITH THE FOLLOWING MEANINGS.
C             LERROR = 0 - NO ERRORS, NORMAL TERMINATION.
C             LERROR = 1 - INVALID INPUT PARAMETER.
C             LERROR = 2 - A DIAGONAL ELEMENT OF THE TRIANGULAR MATRIX IS
C                          ZERO.
C
C
C
C    *** AUTHORS:    MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                    SYSTEMS OPTIMIZATION LABORATORY
C                    DEPARTMENT OF OPERATIONS RESEARCH
C                    STANFORD UNIVERSITY
C                    STANFORD, CALIFORNIA 94305
C
C    *** DATE:       DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C       DECLARATION OF LOCAL VARIABLES.
C
        INTEGER   I, II, ISTART, K, NIM1, NLOOP
        DOUBLE PRECISION   YVAL
C
C       DECLARATION OF STANDARD FUNCTIONS.
C
        INTEGER   IABS
C
C
C
C   TEST FOR ERROR IN INPUT PARAMETERS.
C
        LERROR = 1
        IF (N .LE. 0 .OR. NCOL .LE. 0 .OR. N .GT. NCOL)  RETURN
        IF (IABS(INCSL) .NE. 1)  RETURN
C
C   INITIALIZE THE Y VECTOR TO ZERO.
C
        DO 10 I = 1, NCOL
           Y(I) = 0.0D+0
   10 CONTINUE
C
C-----------------------------------------------------------------------
C
C   SET UP INDICES TO SOLVE EITHER A LOWER TRIANGLE (FORWARD) OR AN
```

```
C  UPPER TRIANGLE (BACKWARD).
C
C-----------------------------------------------------------------------
C
      IF (INCSL .LT. 0)  GO TO 20
      ISTART = 1
      GO TO 30
   20 ISTART = N
   30 CONTINUE
      K = ISTART
      LERROR = 2
C
C-----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 100 RUNS OVER THE UNKNOWNS.  K GIVES THE
C  INDEX OF THE UNKNOWN CURRENTLY BEING SOLVED FOR.
C
C-----------------------------------------------------------------------
C
      DO 100 NLOOP = 1, N
         IF (QRV(K,K) .EQ. 0.0D+0)  RETURN
         YVAL = B(K)
         IF (NLOOP .EQ. 1)  GO TO 60
         NLM1 = NLOOP - 1
         I = ISTART
         DO 50 II = 1, NLM1
            YVAL = YVAL - QRV(K,I)*Y(I)
            I = I + INCSL
   50    CONTINUE
   60    Y(K) = YVAL/QRV(K,K)
         K = K + INCSL
  100 CONTINUE
C
C  THE LOOP HAS TERMINATED NORMALLY.
C
      LERROR = 0
      RETURN
      END
```

6.11. Subroutine TRTSLV

6.11.1. Purpose

Subroutine TRTSLV returns the solution of a non-singular $r \times r$ linear system wherein the matrix is the transpose of the triangular matrix stored in the first r columns of an r by $n (n \geq r)$ matrix. TRTSLV may be useful in solving linear systems related to the orthogonal factorization of a given matrix -- for example, in finding a solution to a set of r linear equality constraints involving n variables.

6.11.2. Description of method

The standard methods of solving triangular systems are used, as in TRSLV. The only notable feature of TRTSLV is that the matrix involved in the linear system is the transpose of the indicated triangular form. Thus, if the original matrix is upper triangular, forward elimination is used in solving its transpose.

6.11.3. Keywords

Triangular linear system; transposed linear system.

6.11.4. Source language

Fortran. The code in TRTSLV has been checked by the PFORT verifier, and is WATFIV-compatible. All variables are explicitly declared.

6.11.5. Specification and parameters

See accompanying listing.

6.11.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.11.7. Auxiliary routines

None.

6.11.8. Program size

37 Fortran source statements.

6.11.9 Array storage

No locally declared arrays.

6.11.10. Timing

The number of arithmetic operations required to solve an r by r triangular system is of order $r^2/2$.

6.11.11. Further comments

None.

```
      SUBROUTINE TRISLV ( N, NDIM, QRV, B, INCSL, Y, LERROR )
C
      INTEGER    N, NDIM, INCSL, LERROR
      DOUBLE PRECISION   QRV(NDIM, N), B(N), Y(N)
C
C
C-----------------------------------------------------------------------
C
C  THE SUBROUTINE TRISLV IS USED TO SOLVE A SYSTEM OF LINEAR EQUATIONS,
C  WHERE THE MATRIX TO BE USED IS THE TRANSPOSE OF A TRIANGULAR MATRIX
C  (LOWER OR UPPER), STORED IN THE UPPER LEFT N BY N CORNER OF QRV.
C  TRISLV IS SPECIFICALLY DESIGNED FOR PROBLEMS ASSOCIATED WITH
C  AN ORTHOGONAL FACTORIZATION OF QRV, WHERE LINEAR SYSTEMS ARISE
C  INVOLVING BOTH THE TRIANGULAR MATRIX AND ITS TRANSPOSE.
C
C  THE FORMAL PARAMETERS OF TRISLV ARE --
C
C  N -
C        INTEGER, INPUT ONLY.
C        THE SIZE OF THE TRIANGULAR MATRIX.
C
C  NDIM -
C        INTEGER, INPUT ONLY.
C        THE DECLARED ROW DIMENSION OF QRV.  MUST BE .GE. N.
C
C  QRV -
C        DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NDIM BY N.
C        THE TRIANGULAR MATRIX USED IN TRISLV IS CONTAINED IN THE UPPER
C        LEFT CORNER OF QRV.
C
C  B -
C        DOUBLE PRECISION VECTOR, OF LENGTH N.  INPUT ONLY.
C        THE RIGHT-HAND SIDE OF THE SYSTEM OF EQUATIONS.
C
C  INCSL -
C        INTEGER, INPUT ONLY.
C        THE VALUE OF INCSL INDICATES WHETHER THE MATRIX IS LOWER OR
C        UPPER TRIANGULAR.  IF INCSL = 1, THE MATRIX IS CONSIDERED TO
C        BE LOWER TRIANGULAR.  IF INCSL = -1, THE MATRIX IS AN UPPER
C        TRIANGLE.  NOTE THAT THE MATRIX USED IN SOLVING THE LINEAR
C        SYSTEM IS THE TRANSPOSE OF THE TYPE OF TRIANGLE INDICATED BY
C        THE VALUE OF INCSL.
C
C  Y -
C        DOUBLE PRECISION ARRAY, OF LENGTH N.  OUTPUT ONLY.
C        THE FIRST N COMPONENTS OF Y ARE THE SOLUTION OF THE
C        TRIANGULAR SYSTEM, AND THE REMAINING COMPONENTS ARE SET TO
C        ZERO.
C
C  LERROR -
```

```
C        INTEGER, OUTPUT ONLY.
C        AN ERROR FLAG, WITH THE FOLLOWING MEANINGS.
C           LERROR = 0 - NO ERRORS, NORMAL TERMINATION.
C           LERROR = 1 - INVALID INPUT PARAMETER.
C           LERROR = 2 - A DIAGONAL ELEMENT OF THE TRIANGULAR MATRIX IS
C                        ZERO.
C
C
C
C   *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                 SYSTEMS OPTIMIZATION LABORATORY
C                 DEPARTMENT OF OPERATIONS RESEARCH
C                 STANFORD UNIVERSITY
C                 STANFORD, CALIFORNIA 94305
C
C   *** DATE:     DECEMBER 1977
C
C
C-----------------------------------------------------------------------
C
C
C      DECLARATION OF LOCAL VARIABLES.
C
       INTEGER   I, II, INC, ISTART, K, NIM1, NLOOP
       DOUBLE PRECISION   YVAL
C
C      DECLARATION OF STANDARD FUNCTIONS.
C
       INTEGER   IABS
C
C
C
C  TEST FOR ERROR IN INPUT PARAMETERS.
C
       LERROR = 1
       IF (N .LE. 0)  RETURN
       IF (IABS(INCSL) .NE. 1)  RETURN
C
C  INITIALIZE THE Y VECTOR TO ZERO.
C
       DO 10 I = 1, N
          Y(I) = 0.0D+0
   10  CONTINUE
C
C-----------------------------------------------------------------------
C
C  SET UP INDICES TO SOLVE EITHER A LOWER TRIANGLE (FORWARD) OR AN
C  UPPER TRIANGLE (BACKWARD).
C
C-----------------------------------------------------------------------
```

```
C
      IF (INOSL .LT. 0)  GO TO 20
      ISTART = N
      INC = -1
      GO TO 30
   20 ISTART = 1
      INC = 1
   30 CONTINUE
      K = ISTART
      LERROR = 2
C
C----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 100 RUNS OVER THE UNKNOWNS.  K GIVES THE
C  INDEX OF THE UNKNOWN CURRENTLY BEING SOLVED FOR.
C
C----------------------------------------------------------------------
C
      DO 100 NLOOP = 1, N
         IF (QRV(K,K) .EQ. 0.0D+0)  RETURN
         YVAL = B(K)
         IF (NLOOP .EQ. 1)  GO TO 60
         NLM1 = NLOOP - 1
         I = ISTART
         DO 50 II = 1, NLM1
            YVAL = YVAL - QRV(I,K)*Y(I)
            I = I + INC
   50    CONTINUE
   60    Y(K) = YVAL/QRV(K,K)
         K = K + INC
  100 CONTINUE
C
C  THE LOOP HAS TERMINATED NORMALLY.
C
      LERROR = 0
      RETURN
      END
```

6.12. Subroutine UNSCRM

6.12.1. Purpose

The subroutine UNSCRM constructs a re-ordered n-vector by applying a specified permutation (or its inverse).

6.12.2. Description of method

Let: iperm(i) denote the i-th component of the permutation; x(i) denote the i-th component of the original n-vector; and y(i) denote the i-th component of the re-ordered vector. If the permutation is applied, the vector y is defined by:

$$y(iperm(i)) \leftarrow x(i), \qquad i = 1, 2, \ldots, n \; .$$

If the inverse permutation is applied, then

$$y(i) \leftarrow x(iperm(i)), \qquad i = 1, 2, \ldots, n \; .$$

6.12.3. Keywords

Permutation; re-ordering.

6.12.4. Source language

Fortran. The code in UNSCRM has been checked by the PFORT verifier, and is WATFIV-compatible. All variables are explicitly declared.

6.12.5. Specification and parameters

See accompanying listing

6.12.6. Error indicators

None. The user is responsible for the consistency of the specified permutation, i.e., each integer between 1 and n should occur only once. The actual parameters corresponding to the original and re-ordered vectors must <u>not</u> be the same.

6.12.7. Auxiliary routines

None.

6.12.8. Program size

18 Fortran source statements.

6.12.9. Array storage

No locally declared arrays.

6.12.10. Timing

The re-ordering requires n indirect addresses.

6.12.11. Further comments

None.

```
      SUBROUTINE UNSCRM ( KFLAG, N, IPERM, XORIG, XPERM )
C
      INTEGER    KFLAG, N
      INTEGER    IPERM(N)
      DOUBLE PRECISION    XORIG(N), XPERM(N)
C
C
C------------------------------------------------------------------------
C
C          THE SUBROUTINE UNSCRM IS USED TO APPLY A PERMUTATION (OR ITS
C     INVERSE) OF ONE VECTOR TO ANOTHER.
C
C     THE FORMAL PARAMETERS OF UNSCRM ARE --
C
C     KFLAG -
C          INTEGER, INPUT ONLY.
C          KFLAG INDICATES WHETHER THE PERMUTATION OR ITS INVERSE IS TO
C          BE APPLIED.  IF KFLAG = 1, THE PERMUTATION ITSELF IS USED,
C          I.E., THE I-TH COMPONENT OF XORIG BECOMES THE IPERM(I)-TH
C          COMPONENT OF XPERM.  FOR EXAMPLE, IF KFLAG = 1 AND
C          IPERM = ( 3 1 2 ), THEN XPERM IS GIVEN BY
C                                     ( XORIG(2) )
C                                     ( XORIG(3) )
C                                     ( XORIG(1) ).
C
C          IF KFLAG = 2, THEN THE INVERSE PERMUTATION IS APPLIED, I.E.,
C          THE IPERM(I)-TH COMPONENT OF XORIG BECOMES THE I-TH COMPONENT OF
C          XPERM.  WITH THE IPERM ARRAY GIVEN ABOVE, AND KFLAG = 2, THE
C          XPERM VECTOR WILL BE GIVEN BY
C                                     ( XORIG(3) )
C                                     ( XORIG(1) )
C                                     ( XORIG(2) ).
C     N -
C          INTEGER, INPUT ONLY.
C          THE NUMBER OF COMPONENTS IN THE VECTOR TO BE PERMUTED.
C
C     IPERM -
C          INTEGER ARRAY OF LENGTH N, INPUT ONLY.
C          THE IPERM ARRAY REPRESENTS THE RE-ORDERING TO BE APPLIED, AS
C          EXPLAINED ABOVE UNDER 'KFLAG'.
C
C     XORIG -
C          DOUBLE PRECISION ARRAY OF LENGTH N, INPUT ONLY.
C          THE ORIGINAL VECTOR TO BE RE-ARRANGED.
C
C     XPERM -
C          DOUBLE PRECISION ARRAY OF LENGTH N, OUTPUT ONLY.
C          THE RE-ORDERED VECTOR.  THE ACTUAL PARAMETERS CORRESPONDING
C          TO XORIG AND XPERM MUST NOT ( REPEAT, NOT ) BE THE SAME.
C
```

```
C
C    *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                  SYSTEMS OPTIMIZATION LABORATORY
C                  DEPARTMENT OF OPERATIONS RESEARCH
C                  STANFORD UNIVERSITY
C                  STANFORD, CALIFORNIA 94305
C
C    *** DATE:     DECEMBER 1977
C
C-----------------------------------------------------------------------
C
C       DECLARATION OF LOCAL VARIABLES.
C
        INTEGER   I, IPER
C
C
        IF (KFLAG .EQ. 2)  GO TO 20
C
C   HERE, THE PERMUTATION AS GIVEN IS APPLIED.
C
        DO 10 I = 1, N
           IPER = IPERM(I)
           XPERM(IPER) = XORIG(I)
10      CONTINUE
        RETURN
C
C   HERE, THE INVERSE PERMUTATION IS APPLIED.
C
20      CONTINUE
        DO 30 I = 1, N
           IPER = IPERM(I)
           XPERM(I) = XORIG(IPER)
30      CONTINUE
        RETURN
        END
```

6.13. Subroutine VMULVC

6.13.1. Purpose

The subroutine VMULVC applies to a vector the sequence of r Householder transformations constructed by HREDR to reduce (from the right) an r by n upper trapezoidal matrix to upper triangular form. VMULVC is used principally in computing the minimum-length least-squares solution.

The sequence of transformations may be applied in either forward or backward order. Let V be the orthogonal matrix which is the product of the transformations as constructed:

$$V = \bar{H}_r \cdots \bar{H}_1 ,$$

where the vector corresponding to $\bar{H}_j$ is zero except in positions j, and $(r + 1)$ through n. VMULVC can apply the transformations in forward order (multiply by V), or in reverse order (multiply by V^T).

6.13.2. Description of method

The transformations are assumed to be stored in compact, normalized form, as described in Section 6.5.11; they are applied in the specified order, taking advantage of the special structure of the transformations. If HVECR(j) is zero, the j-th transformation is skipped.

6.13.3. Keywords

Householder reduction from the right; minimum-length least-squares solution.

6.13.4. Source language

Fortran. The code in VMULVC has been checked by the PFORT verifier, and is WATFIV-compatible. All variables and functions are explicitly declared.

6.13.5. Specification and parameters

See accompanying listing.

6.13.6. Error indicators

See accompanying listing (the description of the parameter LERROR).

6.13.7. Auxiliary routines

VMULVC calls the standard functions IABS and DABS.

6.13.8. Program size

36 Fortran source statements.

6.13.9. Array storage

No internally declared arrays.

6.13.10. Timing

The number of arithmetic operations required is of approximate order $2r(n - r)$.

6.13.11. Further comments

None.

```
      SUBROUTINE VMULVC ( NCOL, NRANK, NDIM, QRV, HVECR, VECIN, MULINC,
     1      VECOUT, IERROR )
C
      INTEGER    NCOL, NRANK, NDIM, MULINC, IERROR
      DOUBLE PRECISION    QRV(NDIM, NCOL) , HVECR(NCOL) , VECIN(NCOL),
     1    VECOUT(NCOL)
C
C
C-----------------------------------------------------------------------
C
C         THE SUBROUTINE VMULVC APPLIES A SPECIAL SEQUENCE OF NRANK
C   HOUSEHOLDER TRANSFORMATIONS TO AN INPUT VECTOR, VECIN, TO YIELD THE
C   VECTOR VECOUT.  THE SEQUENCE OF TRANSFORMATIONS WAS CONSTRUCTED BY
C   THE SUBROUTINE HREDR TO REDUCE THE NRANK BY NCOL UPPER TRAPEZOIDAL
C   PORTION OF THE MATRIX QRV TO UPPER TRIANGULAR FORM.
C
C         LET THE MATRIX V BE GIVEN BY THE PRODUCT OF TRANSFORMATIONS
C
C               V = P(NRANK) * ... * P(2) * P(1),
C
C   WHERE THE VECTOR CORRESPONDING TO THE HOUSEHOLDER TRANSFORMATION
C   P(J) HAS NON-ZEROS IN POSITIONS J AND NRANK+1 THROUGH NCOL, AND IS
C   ZERO ELSEWHERE.
C
C         THE TRANSFORMATIONS MAY BE APPLIED IN EITHER FORWARD OR BACKWARD
C   ORDER, DEPENDING ON THE INTEGER FLAG MULINC.  IF MULINC = +1, THE
C   TRANSFORMATIONS ARE APPLIED IN FORWARD ORDER, I.E.,
C
C             VECOUT = P(NRANK) * ... * P(2) * P(1) * VECIN, OR
C
C             VECOUT = V * VECIN.
C
C
C         IF MULINC = -1, THE TRANSFORMATIONS ARE APPLIED IN REVERSE
C   ORDER, I.E.,
C
C             VECOUT = P(1) * P(2) * ... * P(NRANK) * VECIN, OR
C
C             VECOUT = (V TRANSPOSE) * VECIN.
C
C
C   THE FORMAL PARAMETERS OF VMULVC ARE
C
C   NCOL -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF COLUMNS OF QRV.  MUST BE .GT. 0.
C
C   NRANK -
C         INTEGER, INPUT ONLY.
C         THE NUMBER OF TRANSFORMATIONS TO BE APPLIED.  MUST BE .GT. 0.
```

```
C
C  NDIM -
C        INTEGER, INPUT ONLY.
C        THE DECLARED ROW DIMENSION OF QRV.  MUST BE .GE. NROW.
C
C  QRV -
C        DOUBLE PRECISION ARRAY, OF DECLARED DIMENSION NDIM BY NCOL.
C        INPUT ONLY.
C        THE MATRIX QRV CONTAINS INFORMATION ABOUT THE HOUSEHOLDER
C        TRANSFORMATIONS THAT COMPOSE V.  SEE THE EXTERNAL DOCUMENT-
C        ATION FOR DETAILS.
C
C  HVECR -
C        DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  INPUT ONLY.
C        THE HVECR ARRAY CONTAINS INFORMATION ABOUT THE HOUSEHOLDER
C        TRANSFORMATIONS THAT COMPOSE V.
C
C  VECIN -
C        DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  INPUT ONLY.
C        THE VECTOR TO BE TRANSFORMED.
C
C  MULINC -
C        INTEGER, INPUT ONLY.
C        THE VALUE OF MULINC INDICATES THE ORDER IN WHICH THE TRANSFORMA-
C        TIONS ARE TO BE APPLIED TO VECIN, AS DESCRIBED ABOVE.  IF
C        MULINC = 1, VECOUT = V * VECIN.  IF MULINC = -1, THEN
C        VECOUT = (V TRANSPOSE) * VECIN.
C
C  VECOUT -
C        DOUBLE PRECISION ARRAY, OF LENGTH NCOL.  OUTPUT ONLY.
C        THE TRANSFORMED VECTOR.  THE ACTUAL PARAMETERS CORRESPONDING
C        TO VECIN AND VECOUT MAY BE THE SAME.
C
C  LERROR -
C        INTEGER, OUTPUT ONLY.
C        AN ERROR FLAG, WITH THE FOLLOWING MEANINGS
C           LERROR = 0 - NO ERRORS, NORMAL TERMINATION.
C           LERROR = 1 - INVALID INPUT PARAMETER.
C
C
C  *** AUTHORS:  MARGARET H. WRIGHT, STEVEN C. GLASSMAN
C                SYSTEMS OPTIMIZATION LABORATORY
C                DEPARTMENT OF OPERATIONS RESEARCH
C                STANFORD UNIVERSITY
C                STANFORD, CALIFORNIA 94305
C
C  *** DATE:     DECEMBER 1977
C
C-----------------------------------------------------------------------
C
C     DECLARATION OF LOCAL VARIABLES.
```

```
C
      INTEGER    I, J, JH, NLOOP, NRNKP1
      DOUBLE PRECISION    DOTPRD, FACT
C
C     DECLARATION OF STANDARD FUNCTIONS.
C
      INTEGER    IABS
      DOUBLE PRECISION    DABS
C
C
      IERROR = 1
      IF (NRANK .LE. 0 .OR. NCOL .LE. 0 .OR. NRANK .GT. NCOL)  RETURN
      IF (IABS(MULINC) .NE. 1)  RETURN
      IERROR = 0
      DO 10 I = 1, NCOL
         VECOUT(I) = VECIN(I)
   10 CONTINUE
      IF (NRANK .EQ. NCOL)  RETURN
      NRNKP1 = NRANK + 1
      IF (MULINC .LT. 0)  GO TO 20
      JH = 1
      GO TO 30
   20 JH = NRANK
   30 CONTINUE
C
C-----------------------------------------------------------------------
C
C  THE LOOP TO STATEMENT 100 IS OVER THE TRANSFORMATIONS TO BE APPLIED.
C  JH IS THE INDEX OF THE TRANSFORMATION CURRENTLY BEING APPLIED.
C
C-----------------------------------------------------------------------
C
      DO 100 NLOOP = 1, NRANK
C
C        TEST WHETHER THE TRANSFORMATION WAS SKIPPED.
C
         IF (HVECR(JH) .EQ. 0.0D+0)  GO TO 60
         DOTPRD = HVECR(JH)*VECOUT(JH)
         DO 40 J = NRNKP1, NCOL
            DOTPRD = DOTPRD + VECOUT(J)*ORV(JH,J)
   40    CONTINUE
         FACT = DOTPRD/DABS(HVECR(JH))
         VECOUT(JH) = VECOUT(JH) - FACT*HVECR(JH)
         DO 50 J = NRNKP1, NCOL
            VECOUT(J) = VECOUT(J) - FACT*ORV(JH,J)
   50    CONTINUE
   60    JH = JH + MULINC
  100 CONTINUE
      RETURN
      END
```

7. Acknowledgments

We thank Dr. Michael Saunders for his many helpful suggestions, and the Stanford Linear Accelerator Center for providing computer time.

8. References

Blue, J.L. (1978). "A Portable Fortran Program to Find the Euclidean Norm of a Vector," ACM Trans. Math. Software, Vol. 4, No. 1, pp. 15-23.

Gill, P.E. and Murray, W., eds. (1974). Numerical Methods for Constrained Optimization, Academic Press, London and New York.

Golub, G.H., Klema, V., and Stewart, G.W. (1976). "Rank Degeneracy and Least-Squares Problems," Report CS-76-559, Stanford University.

Kaufman, L. and Pereyra, V. (1978). "A Method for Separable Nonlinear Least Squares Problems with Separable Nonlinear Equality Constraints," SIAM J. Num. Anal., Vol. 15, No. 1, pp. 12-20.

Lawson, C.L., and Hanson, R.J. (1974). Solving Least Squares Problems, Prentice-Hall, New Jersey,

Perold, A.F., and Dantzig, G.B. (1978). "A Basis Factorization Method For Block Triangular Linear Programs," Technical Report SOL 78-7, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California.

Peters, G. and Wilkinson, J.H. (1970). "The Least-Squares Problem and Pseudo-Inverses," Computer Journal, Vol. 13, No. 3, pp. 309-316.

Stewart, G.W. (1973). Introduction to Matrix Computations, Academic Press, New York.

Stewart, G.W. (1977). "The QR Decomposition," in Dongarra, J.J., Bunch, J.R., Moler, C.B., and Stewart, G.W., Preliminary LINPACK User's Guide, Tech. Memo. 313, Applied Math. Division, Argonne National Laboratory.

11.28.78

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SOL 78-8 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) FORTRAN Subroutines to Solve the Linear Least-Squares Problem and Compute the Complete Orthogonal Factorization		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Margaret H. Wright Steven C. Glassman		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0267
9. PERFORMING ORGANIZATION NAME AND ADDRESS Stanford University Department of Operations Research Stanford, CA 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 047-064
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Operations Research Program (Code 434) Arlington, VA 22217		12. REPORT DATE April 1978 13. NUMBER OF PAGES 137
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Linear Equations
Linear Least-Squares
Orthogonal Factorization
QR Factorization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the computational procedures involved in: ~~(i)~~ solution of linear least-squares problems (including systems of non-singular, over- and under-determined linear equations); ~~(ii)~~ formation of the complete orthogonal factorization of a general real matrix. Some aspects of implementation and the estimation of rank are discussed. Full documentation (including source code) is given for a modular set of Fortran subroutines to solve these problems ~~(i)~~ and ~~(ii)~~, and several related problems.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)